GAMA

-Generador Automático de Modelos Animados-

Proyecto de Sistemas Informáticos

Facultad de Informática

Universidad Complutense de Madrid

Autores:
Albert Meco Alías
Daniel Martínez de Leiva Nieto
Marcos Alonso Navarro

Director de Proyecto:
Pedro Antonio González Calero

Los abajo firmantes, Albert Meco Alías, Daniel Martínez de Leiva Nieto, y Marcos Alonso Navarro, autores del proyecto G.A.M.A. (Generador Automático de Modelos Animados), en la asignatura de Sistemas Informáticos, autorizan a la Universidad Complutense de Madrid a difundir y utilizar los contenidos de este proyecto, así como el código, prototipos, o documentación asociada a dicho proyecto, con fines exclusivamente académicos, nunca comerciales, y mencionando expresamente a sus autores.

Participantes:

Albert Meco Alías **Daniel Martínez de Leiva Nieto** **Marcos Alonso Navarro**

Julio de 2006

Resumen del Proyecto

GAMA (Generador Automático de Modelos Animados) es una herramienta que facilita la rápida generación de modelos 3D animados, preparados para un uso inmediato en motores gráficos de última generación. Los modelos, que serán de complexión humana y bípedos, contarán con las animaciones y texturas necesarias que permitan de forma suficiente la diferenciación entre varios personajes generados con esta misma aplicación.

Mediante un interfaz claro e intuitivo, el usuario tiene la posibilidad de configurar un personaje a su medida en pocos minutos, y prepararlo para su posterior exportación a un motor gráfico. De esta forma, GAMA se convierte en una herramienta extremadamente útil a la hora de generar proxies de videojuegos y aplicaciones de entorno 3D, dada la posibilidad de generar en poco tiempo múltiples y variados personajes que cubren cada uno de los recursos gráficos de los que es dependiente una determinada aplicación informática de carácter audiovisual.

Palabras clave: Maya, 3D, personajes, videojuego, poligonal, animación, modelo.

Project Summary

GAMA (Animated Models Automatic Generator) is a tool which makes easy the quick generation of animated 3D models, prepared for a immediate use in last generation graphic engines. The models (bipeds of human build) will count on necessary animations and textures that allow the differentiation between some characters generated with the same application.

With a clear an intuitive interface, the user creates a character in few minutes, and prepares it for it's subsequent export to a graphic engine. On this way, GAMA becomes an extremely useful tool generating videogames' proxies and 3D environment applications, due to the possibility of generating in few time multiple and variated characters, which cover each of the graphic resources needed by an audiovisual application.

Keywords: Maya, 3D, character, videogame, polygonal, animation, model.

INDICE

1. Introducción

En la actualidad, la industria de los videojuegos crece de manera exponencial, y poco a poco ha ido haciéndose notar hasta pasar a ser una de las más lucrativas formas de entretenimiento en el mundo.

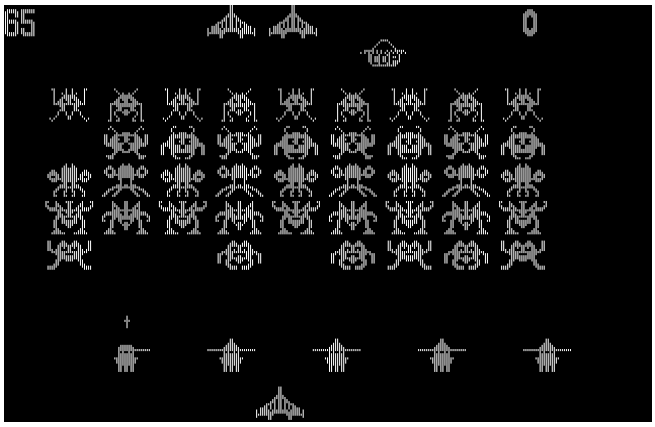
Desde los orígenes del mundo de los videojuegos hasta ahora, ha pasado un largo período de tiempo, y tanto su calidad , como sus objetivos y su forma de producción han variado de manera sustancial.

Existió una época inicial en la que los videojuegos eran el singular entretenimiento de unos pocos. Los programas en sí eran muy simples, así como sus gráficos, y la persona que jugaba esperaba encontrar el entretenimiento instantáneo, esperando una alta y adictiva jugabilidad, sin aspirar por supuesto a encontrar ningún tipo de trama o recurso de ambientación.

Captura de pantalla de Pacman, videojuego de gran éxito de los '80.

En aquella época los programadores de videojuegos eran escasos, y normalmente el equipo de producción constaba solo de uno o dos programadores, y un grafista.

El mercado de entonces no era extenso, y se basaba más en las propias máquinas, caros instrumentos que se adquirían para uso personal (Spectrum, Amiga, Atari), o bien esperando retribución en lugares de ocio (recreativas).



Captura de pantalla del mítico
juego de recreativa Space Invaders (1978).

Desde aquellos tiempos hasta ahora, el mundo de los videojuegos ha ido evolucionando, impulsado por las mejoras tecnológicas, así como por el reconocimiento gradual de este campo del entretenimiento.

Actualmente, los videojuegos mueven miles de millones de euros, promocionados por grandes corporativas de ámbito mundial. Se ha ampliado desde entonces el rango de edad del jugador, y la concepción del videojuego y sus fines han cambiado mucho desde sus principios.

En la época actual los videojuegos constan de una gran complejidad, basándose en trepidantes guiones cinematográficos, y aprovechando una calidad gráfica que mejora cada día. Los juegos tienen bandas sonoras, y se anuncian en spots publicitarios de TV. Mediante el uso de Internet, jugadores de todo el mundo comparten su tiempo jugando en red. Existen ahora videojuegos de todas las categorías, que se adaptan mejor o peor a cada tipo de persona.

Captura de pantalla de juego en tiempo real de Final Fantasy XI, uno de los últimos juegos de gran relevancia, para Playstation2.

Lo mismo ocurre con las plataformas, que cada vez evolucionan más en características como capacidad de proceso, potencia gráfica, tamaño, peso, portabilidad y funcionalidades.

Las plantillas de producción actuales constan de un personal de unos 300 profesionales. Cada uno de ellos está especializado en su tarea, y en la empresa se siguen a rajatabla particulares esquemas jerárquicos y de producción. Existe ya toda una teoría de los videojuegos, adaptada a la perfección a la Ingeniería del Software, y al resto de los múltiples campos que trata el mundo de los videojuegos.

Tanto a nivel sociológico como económico, sería imposible ya prescindir de un campo tan importante hoy en día como el de los videojuegos.

Desde otro punto de vista, el desarrollo de la calidad gráfica en los videojuegos también ha sido notable, y se ha pasado de jugar con unos cuantos pixels que variaban en pantalla en los primeros tiempos, a disfrutar de artísticos y coloridos sprites 2D en la época de los ’90, y por último a interactuar con los más complejos gráficos tridimensionales mezclados con efectos que simulan en alto grado la realidad.

El desarrollo de los gráficos 3D ha definido una nueva escuela, y por tanto una especialidad artística novedosa, y muy mezclada con la informática. Hoy en día existen profesionales del 3D que han dedicado muchos años a obtener la formación adecuada para poder ofrecer los gráficos que se pueden soportar en una plataforma actual.

Es por todo esto que en la actualidad, dentro del ámbito de la programación, el desarrollo gráfico y de videojuegos ha crecido mucho en interés, y de manera lógica ocupa ya un espacio en el mundo de la formación y la investigación. Hoy en día existe un creciente número de proyectos relacionados con este tema en centros de formación como la universidad (Facultad de Informática).

Se da una particularidad en estos últimos casos, y es que el investigador o el alumno poseen los conocimientos necesarios en programación, pero por el contrario carecen de cualquier noción artística, totalmente necesaria sin embargo en el ámbito de los videojuegos de la actualidad.

Se plantea, por tanto, un problema difícil de resolver, ante el déficit de la producción de estos gráficos tan necesarios, y sólo en algunos casos puede recibirse la ayuda remunerada de un profesional, o de alguien con la formación suficiente.

Atendiendo a esta idea, y esperando solucionar este déficit, surge la propuesta de este proyecto, G.A.M.A., una herramienta totalmente accesible para cualquier programador, que le ofrece gráficos 3D dinámicos y configurables, adaptables a cualquier sistema con el que esté trabajando,

y sin ninguna necesidad de conocimientos previos sobre arte o infografía 3D.

2. Descripción general de la aplicación GAMA

Descripción general

En términos absolutos, GAMA (Generador Automático de Modelos Animados) pretende ser una herramienta que facilite la rápida generación de modelos 3D animados, preparados para un uso inmediato en motores gráficos de última generación. Los modelos, que serán de complexión humana y bípedos, contarán con las animaciones y texturas necesarias que permitan de forma suficiente la diferenciación entre varios modelos generados con esta misma aplicación.

La idea principal que envuelve a esta aplicación es que mediante un interfaz claro e intuitivo, un usuario profano en las lides del diseño gráfico –y, en particular, el diseño grafico 3D orientado a videojuegos- tenga la posibilidad de configurar un personaje a su medida en pocos minutos listo para ser exportado a un motor gráfico. De esta forma, GAMA se convierte en una herramienta extremadamente útil a la hora de generar proxies de videojuegos, dada la posibilidad de generar en poco tiempo múltiples y variados personajes que cubran cada uno de los recursos gráficos de los que es dependiente una determinada aplicación informática de carácter audiovisual.

GAMA es una aplicación desarrollada en forma de plug-in, que corre bajo el software propietario Maya 6.0, de Autodesk Alias.

Conocimientos previos para abordar su desarrollo

La tarea de crear un software específico para la generación de personajes animados requiere de ciertos conocimientos específicos sin los cuales la tarea se endurece de forma notable. En primer lugar, se han de tener claro cuales son las necesidades con respecto a recursos gráficos que imperan actualmente en la industria audiovisual, tanto a nivel estético como tecnológico. En segundo lugar, se han de tener conocimientos profundos sobre las tareas que engloban la construcción física de un personaje, esto es, técnicas de correcto modelado poligonal, set up de personajes, proceso de texturizado y animación. Sin conocimientos específicos sobre estos temas, que en cierta medida están encuadraras en mundos más artísticos y más alejados del desarrollo tecnológico propiamente dicho, sería muy difícil abordar el desarrollo de una herramienta de estas características. Por último, son imprescindibles –obviamente- conocimientos de programación en general y de programación en MEL en particular, así como un dominio exhaustivo del interfaz de la aplicación sobre la que corre la aplicación a desarrollar, Maya en este caso.

Sotware

El software necesario para la realización de este proyecto es el siguiente:

Software de programación:
Para poder crear una herramienta interactiva que trabaje con los recursos de Maya, es necesario programar sobre MEL, un lenguaje de script que aprovecha las funciones de maya invocándolas mediante comando, y trabaja con elementos de interface con el usuario.
MEL viene integrado con Maya, y no hay necesidad de instalar ninguna biblioteca adicional para programar con él.

Nota: Ver apartado 3.3.1 para una descripción más detallada de MEL.

Software de edición de código:
Para escribir y editar código MEL hemos utilizado Crimson Editor de, un programa de edición de código fuente de carácter gratuíto, que nos permite diferenciar el código por colores y aporta facilidades a la hora de programar y depurar.

Software de diseño 2D:
Para generar todas las capas de textura para el personaje, hemos utilizado Adobe Photoshop 7.0, software por excelencia de edición fotográfica y diseño 2D, que trabaja con múltiples capas y tiene infinidad de posibilidades.

Software de diseño 3D:
El programa 3D utilizado es Maya, de Autodesk Alias. Decidimos usar este programa debido a nuestro previo conocimiento de todas sus funcionalidades (Los tres componentes del grupo de proyecto hemos dedicado varios años de formación con el programa, recibiendo una amplia formación e incluso impartiendo recientemente cursos del mismo). Además, como hemos dicho antes, Maya incluye MEL, que nos proporciona la unión perfecta entre programación y creación artística.

Software de planificación:
Hemos utilizado Microsoft Project, para gestionar la planificación de nuestro proyecto.

Análisis de requisitos

En el entorno de la programación de videojuegos o demos gráficas pueden destacarse los siguientes prerrequisitos que el programador observa, a la hora de conseguir terminar su trabajo, y que escapan al rango natural de sus conocimientos.

- Se necesitan gráficos 3D, con su consecuente coherencia y estética artística.

- Dentro de estos gráficos, existen normalmente personajes humanos o humanoides.

- Estos personajes deben tener una mínima relación con el resultado que se busca en el proyecto. Por ello, conviene que exista cierta libertad por parte del programador para decidir el aspecto de los personajes.

- En caso de poder influir de manera directa en el resultado final, se espera que los medios sean fáciles e intuitivos, sin aceptar en ningún caso una preparación o investigación previa sobre el mundo de la infografía.

- Dentro de estas particularidades del aspecto general de un personaje, sería bueno el poder definir características básicas como las proporciones, o la textura del mismo.

- Además, cualquier videojuego es una herramienta con gráficos dinámicos, por lo que es necesario que los personajes posean animaciones.

- Sería de agradecer que el juego de animaciones de un personaje sea extenso, y la selección de las animaciones para un personaje en particular pueda ser editable.

- Además de todo esto, es exigible por parte del programador que no existan problemas de compatibilidad de formatos entre el supuesto generador de los gráficos y el motor gráfico utilizado, en caso de existir.

Todos estos prerrequisitos han sido tenidos en cuenta a la hora de llevar a cabo nuestro proyecto, y finalmente se ha intentado mejorar la calidad del mismo, añadiendo ciertas mejoras sobre la base.

3. Revisión detallada de los aspectos principales de GAMA.

División entre trabajo creativo y trabajo de programación.

Para poder desarrollar un proyecto como éste, que aporta soluciones artísticas a los programadores de videojuegos, es evidente que además de las nociones básicas de programación gráfica, son necesarios también amplios conocimientos sobre infografía, y arte en general.

Puede dividirse nuestro trabajo, por tanto, en dos ramas principales:

Trabajo Creativo:

Necesario para generar todas las bibliotecas artísticas de las que hará uso nuestro programa para poder generar los personajes 3D de manera automática.

Abarca los siguientes puntos:

- Amplios conocimientos de algún software de diseño 3D (Maya en nuestro caso), y todas las etapas de trabajo que existen implícitas en la creación desde cero de un personaje animado. En caso de no tener previamente la extensa y compleja formación en el campo especificado en este punto, hubiera sido imposible desarrollar esta aplicación (Al fin y al cabo, para poder eliminar la necesidad del programador de nociones artísticas, ha de cubrirse ésta a un nivel por debajo, es decir, a la hora de implementar G.A.M.A.).
- Modelado desde cero de una malla de personaje básica, así como la generación manual de cada una de las modificaciones posibles que se podrán establecer al personaje (Alrededor de 140).
- Creación de múltiples capas de textura, y sus respectivos mapas de transparencia (Nótese el uso imprescindible de Photoshop para este apartado).
- Generación del set-up de un personaje, para su posterior animación (Incluye la creación del esqueleto, implementación de cinemática inversa para las animaciones, y pesado manual del esqueleto a la malla).
- Creación de animaciones genéricas para el personaje, cuando la biblioteca de animaciones externas no cumplía con las expectativas básicas de nuestro personaje.

Trabajo de Programación:

Como ya se ha dicho antes, el trabajo de programación ha sido desarrollado con MEL (Maya Embedded Languaje).

Son necesarios, por tanto, conocimientos de programación imperativa, y de informática gráfica, unidos al aprendizaje específico del lenguaje de MEL. Dado que hay que invocar por comando las funcionalidades de Maya, es preciso tener un extenso conocimiento de todos los recursos internos que alberga este software, para sacar el máximo provecho.

Existe también trabajo de gestión de ficheros, para trasladar información de unos módulos a otros, y poder guardar y cargar configuraciones y recursos.
Para conseguir características importantes como encapsulamiento, distintos niveles de abstracción del código, transparencia e independencia del código, ha sido aplicada la teoría sobre arquitectura de código e ingeniería del software.

Requerimientos técnicos de los recursos artísticos.

Modelado de una malla inicial.

El proceso de generación de una malla nueva para cada uno de los personajes que podemos obtener con la aplicación GAMA se obtienen a base de infligir deformaciones sobre una malla base. Combinando entre si las distintas deformaciones obtendremos personajes de morfologías muy dispares. Por tanto, para comenzar el proceso de deformación de la malla inicial necesitamos, obviamente, una malla inicial.

La aplicación generará personajes bípedos. Esto nos obliga a contar en un primer lugar con una malla base de rasgos humanos y bípeda. Su construcción no debe dejarse al azar, pues se han de tener en cuenta ciertos aspectos de suma importancia para la correcta deformación de la misma. Es importante reseñar que, en última instancia, estamos tratando con geometría, nubes de puntos que se interconectan entre si, aristas que se comprimen y expanden. Una distribución desafortunada de esta geometría podría hacer imposible la construcción de ciertas deformaciones o la deformación incorrecta de ciertas partes del cuerpo a la hora de animar el modelo generado.

Siendo conscientes de que a priori no conocemos la forma final que tendrá el modelo generado por el usuario mediante la aplicación, tenemos que diseñar la geometría base siguiendo ciertos parámetros que obedecen tanto a razones tecnológicas como aspectos anatómicos particulares del cuerpo humano.

Una primera consideración a tener en cuenta sería establecer como malla base un cuerpo humano bípedo que no destaque físicamente en ninguna de sus partes. Esto es, no tenga, zonas demasiado musculadas o partes concretas del cuerpo especialmente deformadas. La razón de esta consideración es que nuestra malla debe de ser lo más neutral posible, para poder crear a partir de ella deformaciones varias que, por ejemplo, ensanchen la espalda o añadan más musculatura donde sea conveniente. Se intentará construir pues un personaje bípedo básico sin características anatómicas especialmente relevantes.

Sabiendo que la malla será sometida a deformaciones varias, algunas de ellas traumáticas, y que aguantará una mezcla de varias de esas deformaciones en cada momento, no podemos caer en la tentación de generar una malla inicial que cuente con muchos elementos de complicada deformación, como pueden ser triángulos o polígonos de más de cuatro lados. Así mismo, intentaremos evitar la acumulación de aristas en ciertos puntos. Una correcta distribución de los vértices que componen la geometría del muñeco es de vital importancia, pues no solo es fundamental de las futuras deformaciones a las que la malla va a ser sometida, sino que también juega un papel decisivo a la hora de aplicar de forma correcta una textura y para cierto tipo de técnicas de iluminación propias de los motores gráficos, como puede ser la iluminación por vértices.

3.2.1.1. Bloques anatómicos principales.

A la hora de desarrollar las múltiples deformaciones que gestiona nuestra aplicación, hemos de trabajar centrándonos en los principales bloques anatómicos de los que consta un personaje bípedo. A saber, estos son: cabeza – haciendo distinción entre rasgos faciales y deformaciones de cráneo -, torso, brazos, caderas y piernas.

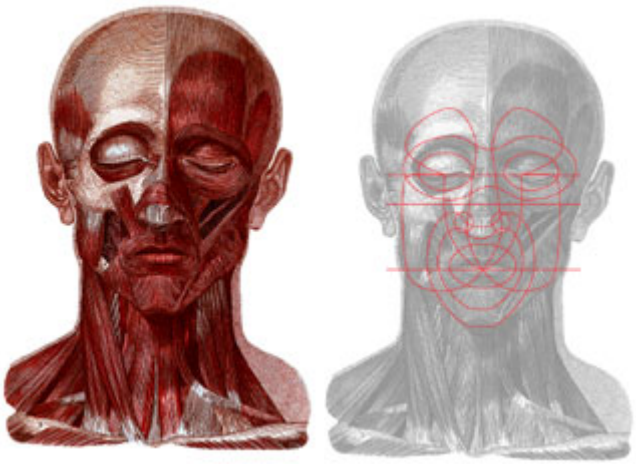
Para cada uno de estos bloques se han generado múltiples deformaciones diferentes entre si. En primer lugar y con la voluntad de poder configurar la envergadura y corpulencia del personaje, se han practicado deformaciones sobre los grupos de músculos más importantes del personaje, pudiendo así agrandar o empequeñecer ciertos músculos de forma individual. Adicionalmente, se han creado deformaciones más específicas para poder caracterizar de una forma más concreta un personaje, añadiendo así la posibilidad de incorporar varios tipos de botas, de zapatos, tipos de vestimenta como camisetas, pantalones, diferentes tipos de peinados, cuernos y un largo etcétera que, mediante una meditada combinación, permiten generar personajes muy dispares.

Asi mismo, se ha creado un juego bastante amplio de deformaciones aplicables a los rasgos faciales, con el fin de configurar la expresión y fisonomía del personaje.

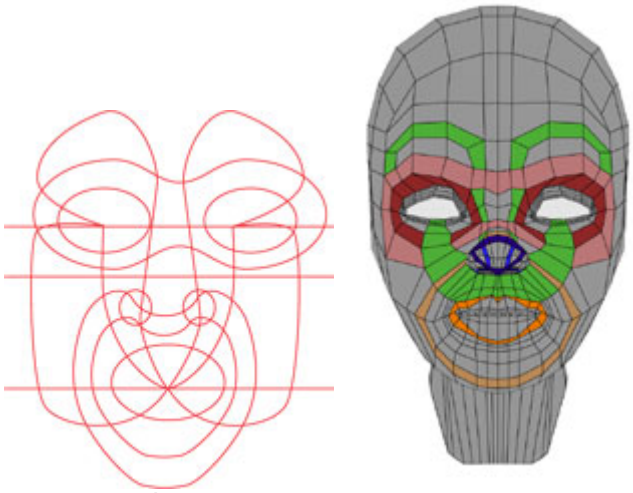
El número de deformaciones combinables en la aplicación G.A.M.A. gira en torno a 140.

3.2.1.2. Teoría de los edge loops.

La técnica del uso de edge loops ha evolucionado paralelamente a la capacidad de computación de los ordenadores actuales. Los primeros modelos en 3D tenían una baja cantidad de polígonos, generando una apariencia poco real y mostrando una triangulación muy evidente, postrando así a la animación a un juego pobre y robótico en cuestión de movimientos. A medida que aumentó la capacidad de computación, aumentaron el número de polígonos que pueden soportar los modelos para videojuegos, adquiriendo de esta forma los modelos más realismo. El problema viene a la hora de deformar una masa de polígonos para que sigan mostrando una parte del cuerpo sin que se evidencie la existencia de polígonos. Para solventar este problema, se intenta desplazar las lineas de polígonos en relación con las formas que describen en la realidad los músculos. Edge loop es una es una técnica pues de modelado cuya máxima premisa es que las aristas de un modelo deben seguir en forma cíclica (LOOPs) imitando a las fibras de los músculos.



Aquí podemos ver las diferentes regiones musculares presentes en la region facial y los loops que describen.



Ejemplo práctico de modelado en el que se ha respetado la distribución de la geometría en base a los loops que sugieren los grupos de músculos.

Cuando movemos nuestros cuerpos, decimos que tensionamos o relajamos músculos. La idea de relajar o tensar están presentes en las dos etapas del modelo 3D cuyo edge loop nos auxilia perfectamente. La primera etapa es la de la tensión poligonal que debemos controlar en la construcción del personaje. La segunda es la de tensión de movimiento propiamente, donde los edge loops ayudan a la correcta deformación de los polígonos. Esto es muy importante a la hora de conseguir una correcta deformación de la superficie poligonal a la hora de animar el modelo.

El concepto de tensión poligonal hace referencia a la acumulación de geometría o mala construcción de la misma en ciertas zonas, rompiendo así la estructura de los edge loops. La tensión poligonal, a parte de no dejar una malla con una topología adecuada para su deformación, puede presentar problemas técnicos a la hora de mostrar nuestro personaje en un motor 3D, como veremos más adelante.

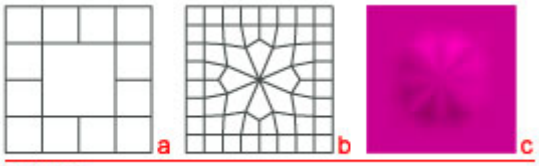


Figura 1

Vemos como una distribución anárquica de la geometría produce un suavizado de la superficie no demasiado elegante.

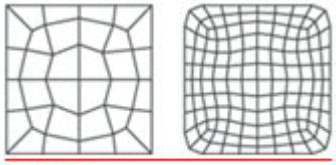


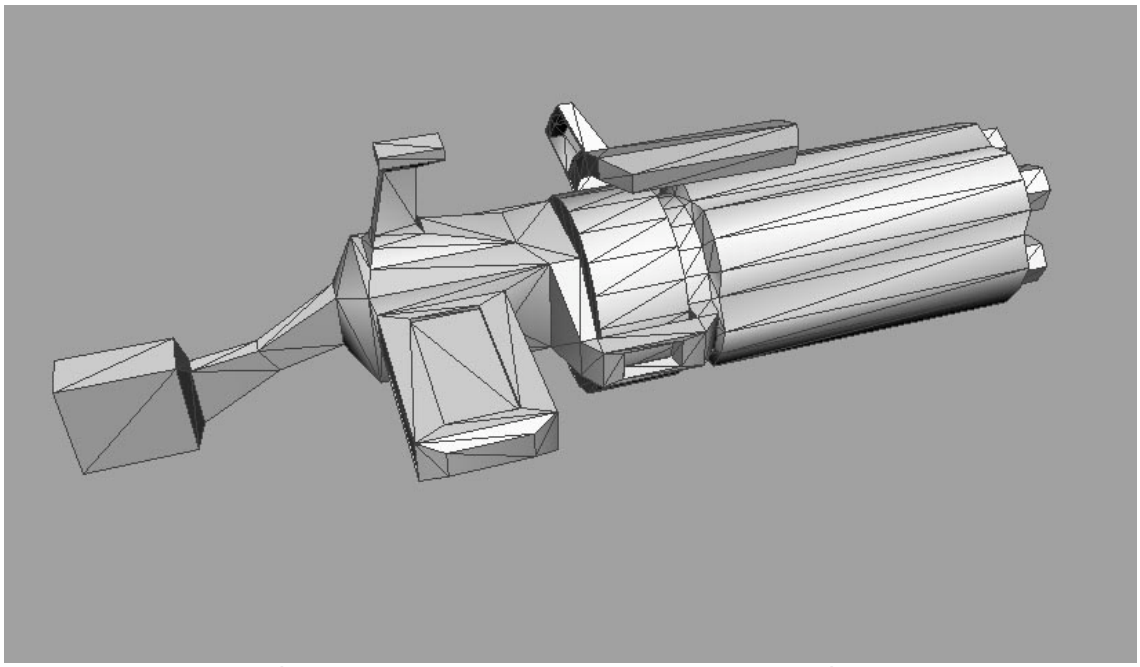
Figura 2

Con una distribución mas coherente de la geometría Eliminamos esos problemas

Para el correcto uso de los edge loops, es importante ciertos conocimientos de anatomía, pues su uso está íntimamente ligado con la estructura del sistema de músculos del personaje que se esté creando.

Donde un personaje no presenta movimientos como la nuca, orejas, antebrazo de un personaje cartoon, dientes, etc., los loops no son necesarios. En esos mismos puntos, el surgimiento de polígonos de 5 o 3 lados no damnifican al personaje sin embargo tener cuidado de usarlos mínimos posibles siempre ayuda.

Los muebles y escenarios en general no necesitan edge loops, pues por regla general no necesitan ser animados imitando movimientos orgánicos. Es ésta, por tanto una técnica para personajes y modelado orgánico.



Triangularización de una malla correspondiente a un objeto no orgánico para videojuegos. Podemos observar como la distribución de la geometría no obedece a ningún tipo de looping.

3.2.1.3. Importancia de la topología en los motores gráficos.

Un personaje animado es, en síntesis, un conjunto de vértices interconectados entre si que conforman una malla, y que se mueven en el espacio con respecto al tiempo. Pese a lo que podría parecer en un principio, las diferencias entre la topología de dos mallas diferentes tienen una importancia notable a la hora de ser representadas en un determinado motor gráfico. Es por esto que tenemos que tener cuidado en ciertos aspectos referentes a la construcción de una malla.

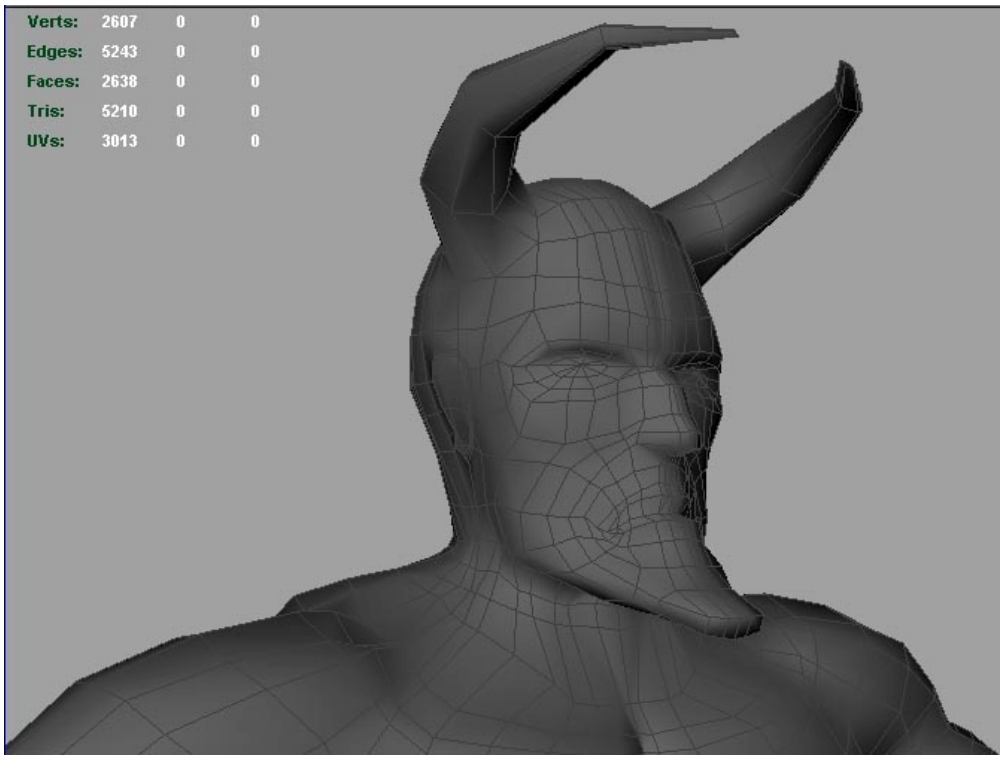
En primer lugar, un dato a tener muy en cuenta es el número de polígonos usados. Técnicamente, lo que realmente nos interesa es el número de triángulos del que cuenta cierto modelo. A priori, cuanto más bajo sea este número, con más agilidad mostrará el motor gráfico nuestro modelo. Es por ello que, en la medida de lo posible, habremos de intentar usar el menor número de polígonos y, por ende, de triángulos.

Dependiendo de si vamos a mostrar de forma simultanea muchos personajes o pocos, las restricciones respecto al número de triangulos variarán. Un personaje que esté diseñado para verse muy pequeño y a una notable distancia, es probable que no necesite más de 200 triángulos. Un personaje estandar sin excesiva resolución puede situarse en torno a los 1500 y 2000 triángulos. Para los motores de nueva generación, se crean personajes que sobrepasan ampliamente los 9.000 triángulos.

En nuestro caso buscaremos una solución de compromiso que tenga en cuenta dos aspectos:

a) Utilizar el menor número de triángulos posibles a fin de el personaje no consuma demasiados recursos sobre la aplicación y máquina en la que vaya a ser utilizado. Siendo ésta una aplicación de propósito general, en la que no estamos ceñidos a un motor gráfico y máquina concretos, intentaremos reducir al máximo el número de polígonos usados siempre que nos sea posible, a fin de favorecer la compatibilidad de nuestros modelos con la mayoría de plataformas existentes.

b) Utilizar la geometría necesaria que permita las múltiples deformaciones de la malla. Para un modelo final concreto, los diseñadores gráficos optimizan de forma manual la malla a fin de reducir el número de triángulos. En nuestro caso, la malla contará con más triángulos de los estrictamente necesarios a fin de que la misma sea fácilmente deformable y permita una amplia variedad de personajes.

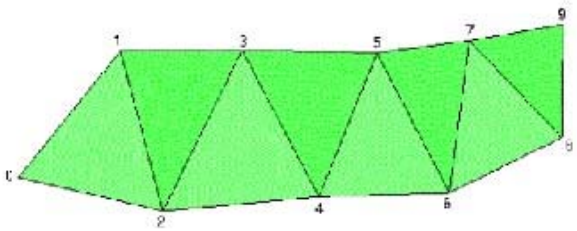


Captura del contador de polígonos sobre nuestra malla base.

Nuestra aplicación generará personajes que tendrán en torno a los 5200 triangulos.

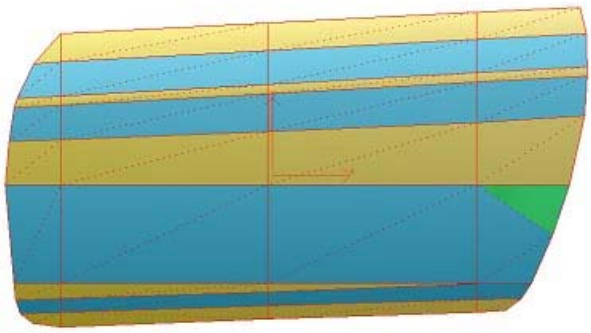
Con respecto a la construcción de la malla, es importante tener en cuenta ciertos conceptos importantes a la hora de crear un personaje. En principio, como ya se explicó cuando habábamos de las bondades de los edge loops, se intentará crear una malla compuesta por polígonos de cuatro aristas. La importancia de este tipo de construcción, a parte de numerosas ventajas a la hora de realizar el pesado y set up de la malla con respecto a la jerarquía de huesos, tiene su importancia también en el momento del renderizado, llevado a cabo por el motor gráfico. Para entender esto último, introduciremos varios conceptos:

- Tri-Strips (tiras de triángulos): Una tira de triángulos es aquella hecha por polígonos –tris- donde cada triángulo individual comparte dos vértices con cada uno de los polígonos vecinos. Todos los polígonos de esta tira comparten las mismas propiedades de material (shader).



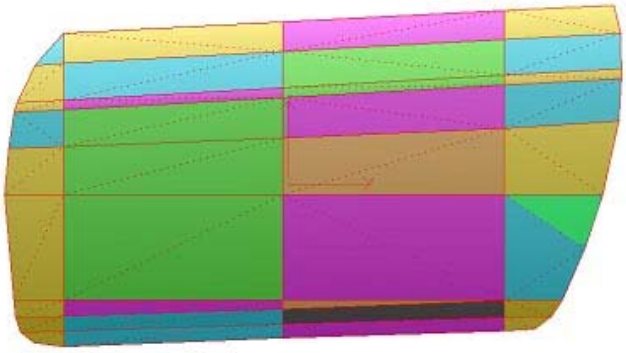
- *Ejemplo de una tira de triángulos.*

  Las tiras de triángulos pueden basarse en polígonos que comparten vértices material común.



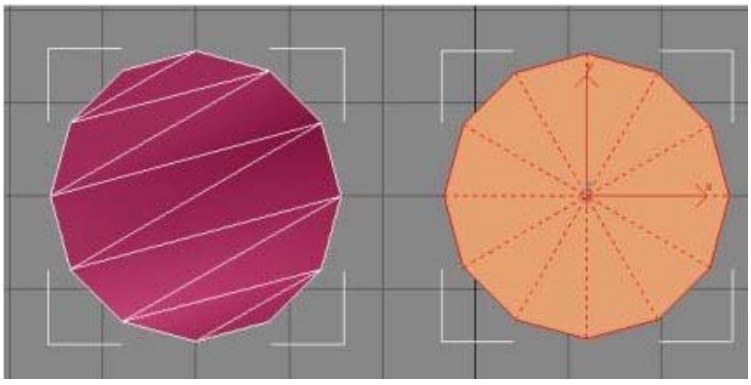
Ejemplo de tiras de triángulos compartiendo Material común

Si se cambian las normales en un edge (borde) y separamos los vértices, quebraríamos la tira. Esto afecta a la calidad y al rendimiento a la hora de renderizar. En relación al tri-stripping intentaremos siembre crear bordes suaves.



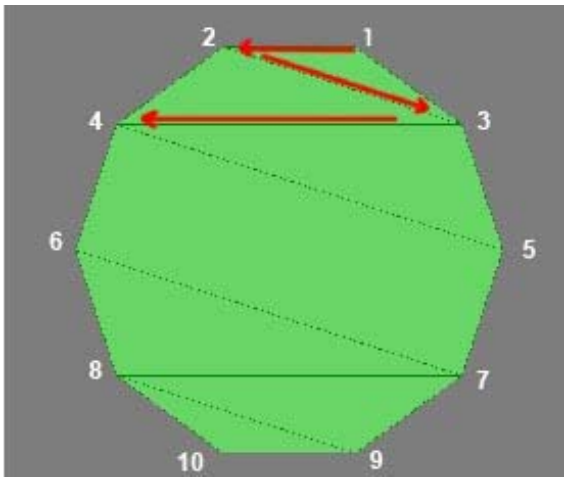
Ejemplo de geometría con alteración en las normales del borde.

- Triangle fan (efecto ventilador): Situación en que múltiples aristas confluyen en un mismo vértice. El sistema de rendeo tarda más en procesar el 'ventilador' que la misma figura dividida en triángulos. Es mejor tratar de dejar la figura como aparece en la parte izquierda del gráfico.



Compracion entre una triangulación eficiente y otra con 'efecto ventilador'

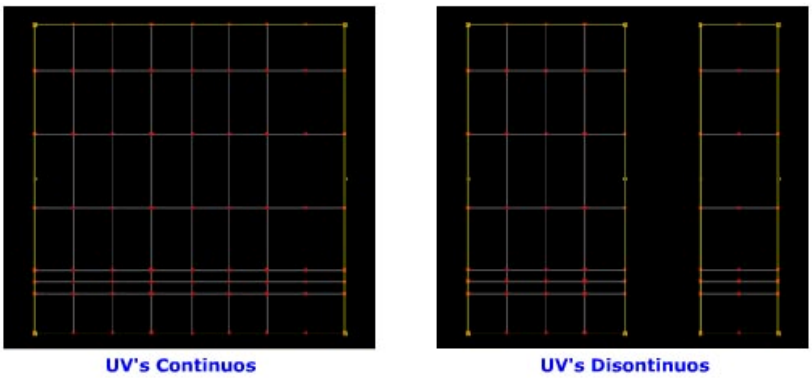
- Cuads libres: Polígonos cuadrados que no han sido triangulados previamente en fase de modelado. En la medida de lo posible, es mejor dejar los polígonos cuadrados sin triangular. Esto da la oportunidad al proceso de rendeo de decidir cual es la mejor manera de desplegar el borde resultante. Esto se conoce como ‘stripper’. El stripper que hace el sistema de rendeo trata de observar las caras adjuntas y separa el cuad a lo largo del eje que mejor preserve la integridad de la tira de triángulos resultante.



Ejemplo de cómo un motor gráfico eficiente triangulalos cuads libres

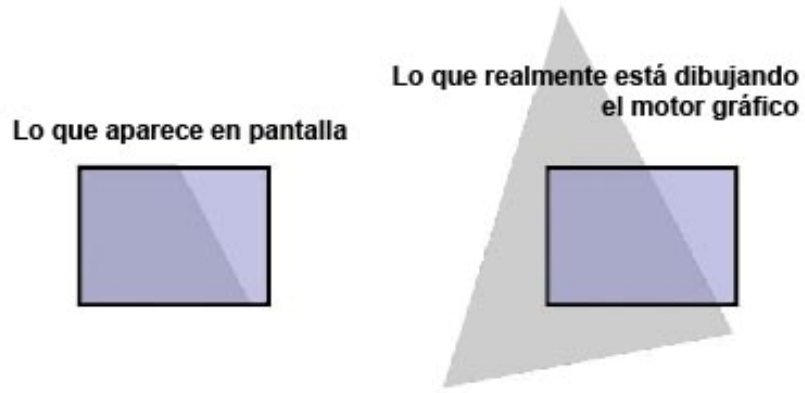
El sistema de rendering va dibujando como lo muestra la figura. Primero dibuja los trazos entre el vértice 1 y 2, luego entre el 2 y el 3 y así sucesivamente. Esto es el rendeo eficiente de geometría en tiempo real. Si tenemos, por ejemplo, un ventilador, el módulo de render va a tardar más tiempo en procesar esa información.

Como reglas básicas del rendeado por los motores gráficos en tiempo real, sabemos que las tiras de triángulos que comparten el mismo punto de pivote pueden ser rendeadas de una sola pasada. En este aspecto, es así mismo importante tener en cuenta las coordenadas de mapeado UV de la malla. En ocasiones, al desplegar el mapeado de la malla, creamos discontinuidades de la misma en algún vértice compartido. Esta discontinuidad entre caras fuerzan a un quiebre a la hora de rendear un determinado tri-strip, con la consiguiente penalización en tiempo de render.



Ejemplo de UVs continuos y discontinuos

De la misma forma, debemos evitar el uso de polígonos 'grandes'. Los motores gráficos, por regla general, solo dibujan en pantalla 'aquello que se está viendo'. No obstante, si solo se visualiza un porcentaje de un determinado polígono en pantalla, el motor se ve obligado a dibujar el polígono entero por hardware, con la consiguiente penalización en tiempo de render.



Efecto de renderizar polígonos de gran dimensión

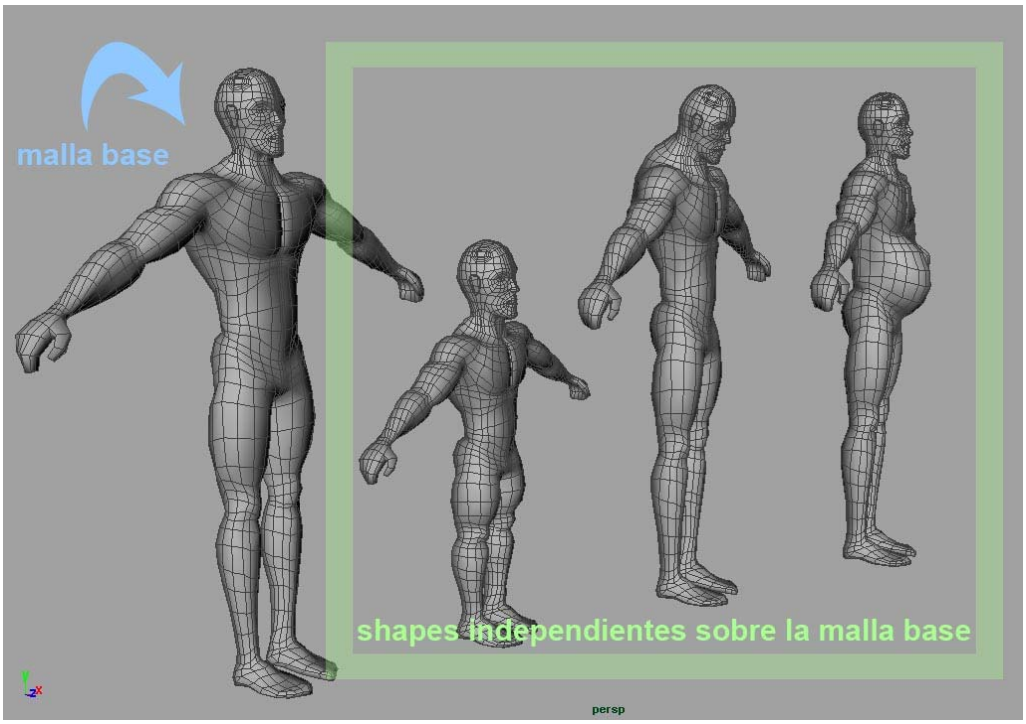
Otro detalle a tener en cuenta a la hora de crear una malla para un personaje, es intentar que los vértices estén distribuidos de una manera lo más uniforme posible a lo largo de la malla. Si se utiliza una iluminación por vértice, la existencia de vértices muy distanciados y otros muy juntos puede generar efectos de iluminación desagradables o no deseados.

Creación de deformaciones de la malla base.

En este capítulo realizaremos una distinción de dos aspectos fundamentales. Por un lado, habremos de realizar un exhaustivo análisis de cuáles son aquellas deformaciones más interesantes a poder combinar partiendo de nuestra malla base, de tal forma que nos permita generar mediante la permutación de las mismas una serie de personajes totalmente diferentes o suficientemente diferenciados. Para ello será necesario analizar qué tipos de personajes queremos poder llegar a realizar y qué requerimientos tienen los mismos.

Por otro lado, tendremos que plantearnos el cómo articular el sistema de deformaciones para generar las mismas y poder hacer que nuestra aplicación las gestione sin problemas.

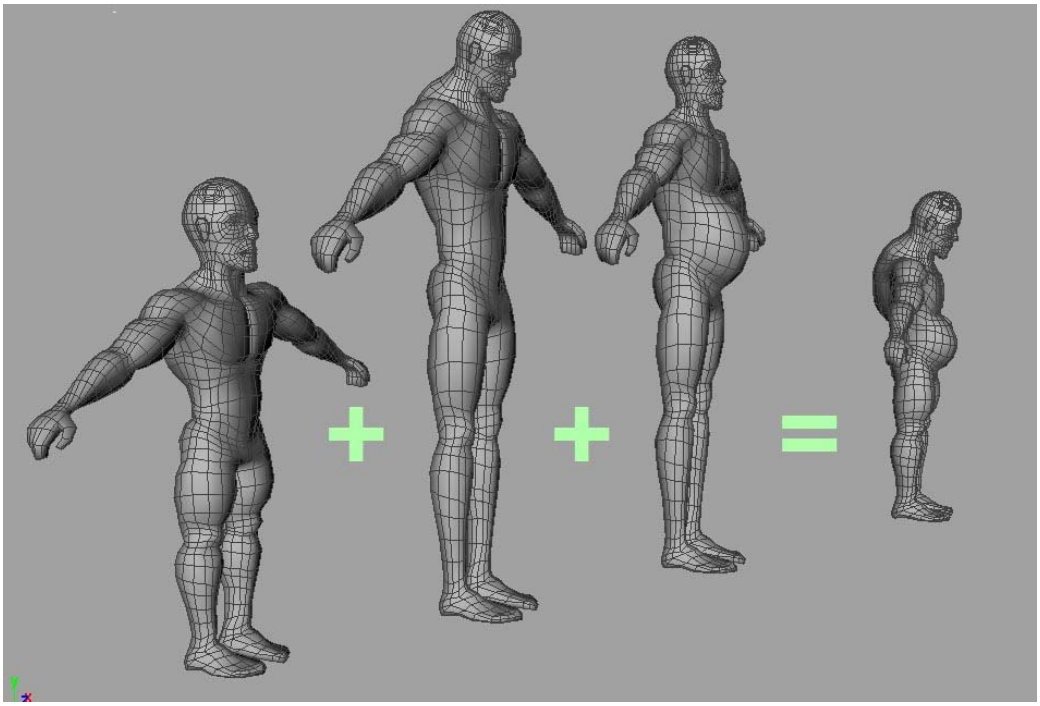
Para implementar en nuestra aplicación las diferentes deformaciones, usaremos la herramienta 'blend shape' propia del software sobre el que la aplicación, en este caso Maya. Una blend shape es una deformación progresiva de la malla, que podemos controlar numéricamente. Para crear una blend shape, partimos de nuestra malla inicial. Duplicamos esta, y sin añadir ningún tipo de geometría, deformamos la malla copia, creando sobre ella la deformación que queramos aplicar sobre la malla original. Es importante que tanto malla original como duplicada cuenten en todo momento con la misma geometría, así como la misma numeración de vértices. Una vez creada la deformación en la malla copia, emparentamos la malla original con la malla copia, creando un nodo 'blend shape' mediante el cual podremos controlar la deformación de forma gradual. Así, por medio de este controlador y partiendo de la malla inicial, podemos aplicar la deformación sobre la malla. El principio matemático sobre el que se sustenta este concepto se basa en interpolar las posiciones de cada vértice teniendo en cuenta la posición inicial del vértice en la malla y la posición del mismo vértice en la malla en la que hemos esculpido la shape, ponderando esta última con el porcentaje de aplicación de la shape.



Conjunto de deformaciones a partir de la malla base.

El control de las deformaciones será transparente al usuario, pues se realizará mediante diversos controladores integrados en la propia aplicación. Mediante programación, se enlazan los controladores a los nodos blend shape de cada deformación.

Es importante hacer notar que cada blend shape funciona paralelamente al resto, con lo que podemos ‘mezclarlas’. Si activamos dos blend shape a la vez sobre una misma zona, la interpolación de las posiciones globales de los vértices se harán con respecto a las posiciones de los mismos en ambas shape y sus porcentajes de influencia –esto es, el peso que le hayamos introducido en cada deslizador-
Cada shape por separado mantiene la coherencia con el modelo, está correctamente modelada y crea una deformación coherente sobre el personaje bípedo. Sin embargo, la explosión combinatoria de mezcla de deformadores podría producir malformaciones en la topología de la malla, o partes del cuerpo de dudosa coherencia física. Esto es imposible de prever, aun así, se intenta que las deformaciones iniciales sean lo más sutiles y naturales posibles, para que su mezcla no de lugar a malformaciones en el dummy inicial.



Ejemplo de interpolación de varias deformaciones.

Creación de la jerarquía de huesos para el control del personaje.

3.2.3.1. El esqueleto humano y su analogía con la jerarquía de huesos del modelo animado.

El esqueleto humano está formado por huesos, algunos de ellos articulados entre sí, soportados por estructuras complementarias como ligamentos, tendones, músculos y cartílagos. Este mismo principio rige la creación de un esqueleto para articular una malla 3D y proporcionar los mecanismos correctos para su posterior animación. Crearemos una estructura de huesos emparentados entre si. Existirá una relación directa entre hueso y malla. Un

vértice estará influido por a lo sumo dos huesos, y su posición en el espacio vendrá determinada por la rotación de dichos huesos.

El esqueleto de un ser humano adulto tiene, aproximadamente, 206 huesos, sin contar las piezas dentarias. Para nuestro fin, es impracticable el uso de tantos huesos, pues no queremos simular a la perfección el comportamiento de un humano en nuestro modelo, sino crear la ilusión de que nuestro modelo se mueve como lo haría un humano. Puesto que nuestro sistema de musculación es mucho menos complejo que el de un humano y nuestro modelo está 'hueco', es decir, no cuenta con masas musculares reales para simular de forma realista las deformaciones producidas por los mismos, nuestro esqueleto será mucho más simple que el humano.

Los huesos tienen tres funciones principales sobradamente conocidas: actúan como sostén de nuestro cuerpo y permiten que este se mantenga erecto, protegen las vísceras ante cualquier presión o golpe del exterior, como, por ejemplo, las costillas al albergar los pulmones, tan delicados y que precisan de un espacio para ensancharse; y además, permiten el movimiento de las extremidades, funcionando como puntos de anclaje de los músculos, que si no los tuvieran no podrían contraerse. En nuestro caso, el esqueleto tendrá como única función articular la malla base. Así mismo contará con ciertos mecanismos adicionales que interactuaran de forma muy estrecha con el esqueleto para facilitar las labores de animación.

3.2.3.2. Puntos de rotación necesarios para articular la malla.

A la hora de construir un esqueleto para animar una malla de modelado orgánico, es necesario llegar a una solución de compromiso entre:

a) Intentar usar el menor número de huesos posibles, a fin de que la labor de set-up y animación sea lo menos tediosa. El número de huesos es importate a la hora de exportar ciertos modelos a determinados motores gráficos, pues algunos de ellos soportan un límite máximo de huesos asignados a una malla.

b) Usar los huesos necesarios para que la articulación de la malla resulte a nivel visual convincente y permita la correcta movilidad de la misma para las labores de animación.

En virtud de estos dos aspectos, se ha determinado necesaria la creación de los siguientes puntos de articulación sobre la malla:

- Base del cuello.
- Base de la cabeza.
- Tórax.
- Omoplatos.
- Hombros.
- Codos
- Muñecas.

- Giro de torso.
- Hueso raíz del muñeco –hueso padre del que derivarán los restantes huesos y que servirá para desplazar al mismo-
- Articulación de las caderas.
- Rodillas.
- Tobillos.

Siguiendo con la filosofía de prototipado de personajes para videojuegos, fin último de esta aplicación, se ha huido de la articulación compleja de partes del cuerpo como pudieran ser los dedos de las manos y juego de tobillo. La decisión en este sentido viene dada por el hecho de que el juego de animaciones que incorporará la aplicación será de propósito general, sin ahondar en animaciones espectaculares y demasiado específicas que, si bien son las que se desarrollan para personajes concretos en aplicaciones comerciales, no se adaptan al concepto de animación estándar para el prototipado de aplicaciones.

Además, este juego simple de puntos de rotación nos permitirá adaptar nuestra jerarquía de huesos a cualquiera de los estándares de animación por captura de movimiento existentes en el mercado (como veremos más adelante), a fin de poder usar las múltiples animaciones existentes en éste formato en el mercado.

3.2.3.3. El formato BVH.

Siguiendo la filosofía de articulación de la malla expuesta en el capítulo anterior, hemos decidido adaptar la jerarquía de huesos a uno de los estándares que podemos encontrar actualmente en el mercado: el formato de captura de movimiento BVH – BioVision Hierarchy –.

El uso de este estándar de animación por captura de movimiento nos permitirá usar ciertas animaciones disponibles en el mercado –bien sean de pago o bien de forma gratuita-. También, y en caso de que el usuario de la aplicación lo considerase oportuno, podría añadir las animaciones que creyese convenientes haciendo uso del kit de captura de movimiento disponible para este estándar.

Así mismo, es importante hacer notar que el uso de este estándar de animación por captura de movimiento no limita la animación del personaje usando exclusivamente métodos de captura de movimiento. Con una configuración avanzada del set-up de la malla, nuestro personaje estaría perfectamente preparado para ser animado por un animador tradicional.

Ejemplo de equipamiento de captura de movimiento.

El formato BVH fue desarrollado originalmente por Biovision, compañía especializada en captura de movimientos. Su desarrollo se iniciaba como respuesta a la necesidad de los clientes de poder manejar de forma eficiente los datos provenientes de la captura de animación.

3.2.3.4. Jerarquía de huesos y disposición en el espacio.

Partiendo de los puntos de rotación básicos mencionados en apartados anteriores, la disposición de huesos en torno a nuestro personaje quedaría de la siguiente manera:

Se añaden huesos cuya animación no será relevante en las partes finales de la jerarquía del esqueleto – final de brazos, final de piernas y fin de cabeza- Estos huesos son de gran utilidad visual a la hora de configurar la animación. En el caso de la muñeca, por ejemplo, la rotación de la misma estará ligada al hueso 'muñeca'. Si no existiese el hueso de fin de brazo, no existiría así mismo el segmento que une la muñeca con dicho hueso. Este segmento nos es de gran utilidad para ver qué dirección tiene la muñeca en cada momento. En situación análoga se encuentran los huesos de los tobillos y el hueso de la cabeza. Es por esto que añadimos dichos huesos de fin de extremidad.

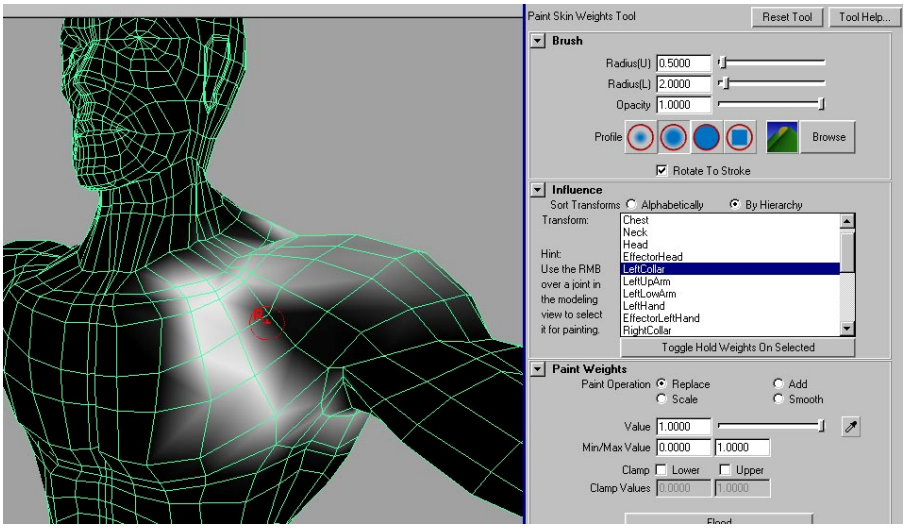
La colocación de los huesos con respecto a las diferentes mallas que pueda configurar el usuario es un aspecto de suma importancia a la hora de realizar la preparación de un personaje para ser animado. En base a una colocación inicial estudiada sobre el personaje básico, y habiendo comprobado que dicha colocación satisface todas las exigencias de animación del personaje, se han seleccionado ciertos vértices maestros que flanquean cada hueso en su parte anterior y posterior. Dichos vértices serán usados como testigos para colocar en los distintos tipos de personajes cada uno de los huesos. Así, haremos una media aritmética de la posición de dichos testigos para determinar dónde habrá de colocarse cada uno de los huesos.

Así pues, a cada punto de rotación corresponde una tupla de vértices que determinarán en qué posición se ubica dicho punto de rotación. La elección de dos y no más vértices como referencia es una decisión arbitraria. Podrían usarse más vértices de control, pero en este caso consideramos suficiente el uso de dos testigos, pues la ubicación de puntos de rotación sobre la malla usando este sistema ofrece soluciones más que aceptables en la mayoría de los modelos susceptibles de ser generados con esta aplicación.

Asignación de la jerarquía de huesos a la malla.

Para posibilitar la animación de un personaje, es imprescindible asociar cada vértice de la malla del mismo con uno o varios huesos de nuestro esqueleto. Así, la posición de un vértice se verá influenciada por los giros efectuados en aquellos huesos que estén emparentados con ese vértice.

La asignación de huesos a una malla o 'pesado de la malla' no es una tarea mecánica, ni fácilmente automatizable. El pesado correcto de una malla es un aspecto fundamental de la preparación de personajes para su correcta animación. De ella depende que los vértices de los que se compone un personaje se muevan de forma natural o que, por el contrario, produzcan deformaciones no deseadas que echen por tierra la credibilidad del mismo. Es pues, una labor de suma importancia la del pesado.



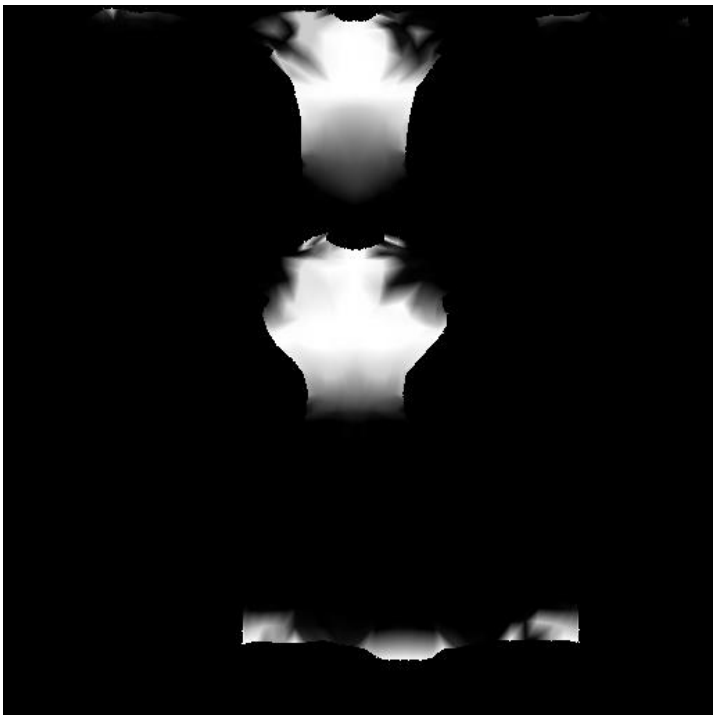
Ejemplo de pesado de un hueso. La parte de color blanca Corresponde a la zona de influencia sobre la malla del hueso que estamos pesando.

Tradicionalmente, los 'riggers', especialistas en pesado y set-up de personajes de grandes estudios de animación, observan minuciosamente como se comporta la piel de modelos reales (humanos, animales, etc) para obtener datos que después aplicarán a sus labores de set-up. Es pues la tarea de 'skinning' una tarea eminentemente manual y de difícil automatización.

Sabemos que nuestra malla siempre cuenta con el mismo número de vértices, pues estos se desplazan en el espacio para configurar la estructura del personaje, pero no varían en número. Así pues, se ha realizado un pesado manual con un minucioso cuidado, poniendo especial atención en las zonas en las que más se compromete la deformación del personaje (articulaciones, zonas de pliegues corporales, etc), intentando realizar una labor de pesado que se adapte a la gran mayoría de topologías de malla. Una vez realizado esta labor de pesado, se ha probado con varias topologías muy diferentes entre si. Una vez encontrado un pesado que se adapta de forma razonable a las topologías más extremas que genera la aplicación, guardamos el mismo en un mapa de pesado.

Un hueso puede influenciar porcentualmente a un vértice. En nuestro sistema, estableceremos como restricción que un vértice pueda ser influenciado por, a lo sumo, dos huesos diferentes. Este es un detalle muy importante a tener en cuenta, pues dependiendo del motor gráfico al que vayamos a exportar nuestro modelo animado, admitirá un mayor o menor número de influencias de hueso por vértice. En este caso, consideramos que dos huesos por vértices es un número de influencias razonable, soportado por la mayoría de los motores gráficos actuales y mínimo de influencias necesario para poder generar animaciones con una fluidez aceptable.

Un mapa de pesado es una colección de imágenes en escala de grises que indican la influencia que ejercerá cada hueso a cada vértice. Habrá tantas imágenes como número de huesos.

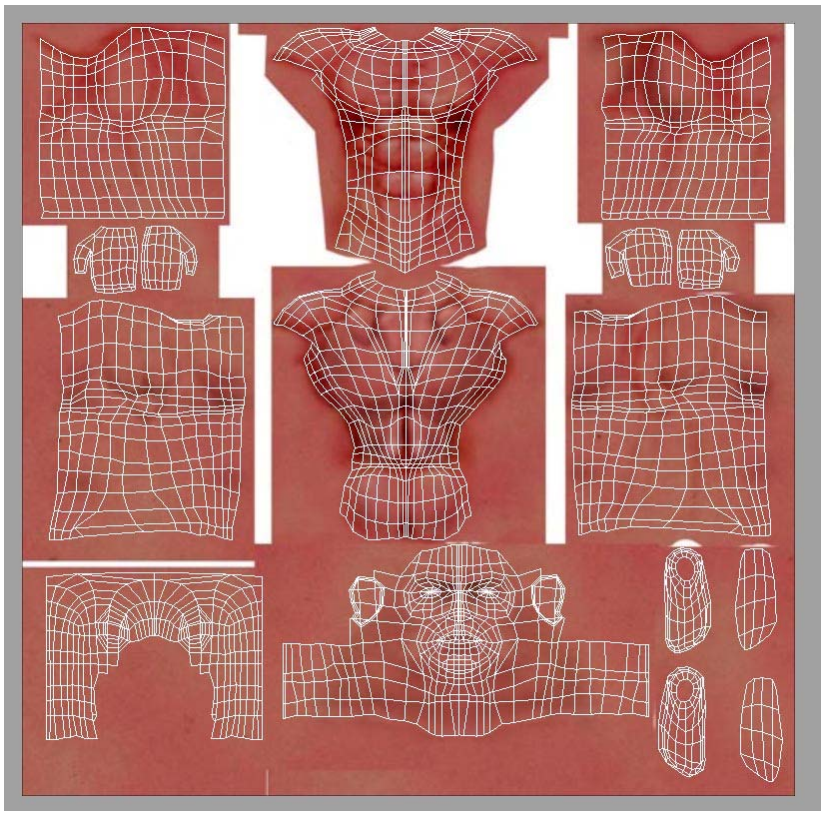


Ejemplo de mapa de pesado. Las zonas de color blanco corresponden a las zonas de influencia del hueso al que corresponde el mapa sobre la malla.

En la ejecución de nuestra aplicación, cada vez que se ejecuten las labores de pesado del personaje se cargará de forma automática este mapa.

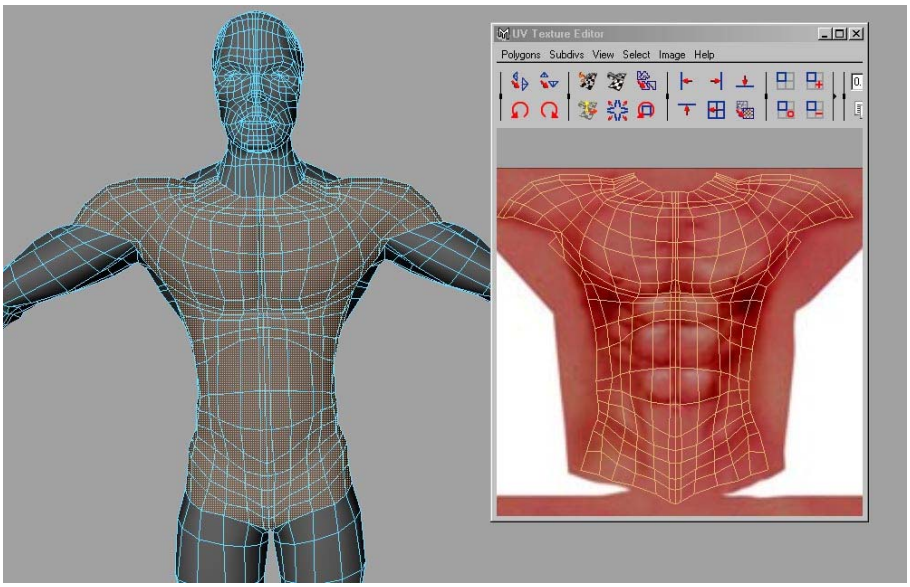
Mapeado de la malla y texturas.

El mapeado de una malla (mapa de UVs) consiste en la asignación de una textura 2D a las caras de un modelo 3D siguiendo las coordenadas de la superficie (las coordenadas UV precisamente). Es uno de los métodos más populares y efectivos para texturizar personajes animados, pero a la vez es uno de los más difíciles y tediosos de utilizar cuando se quieren obtener resultados dignos.



Coordenadas de mapeado del personaje base. Podemos ver cómo la textura se adapta a las coordenadas UVs.

La técnica de mapeado tradicional consiste en seleccionar las diferentes caras poligonales de nuestro modelo e ir haciendo sucesivas proyecciones planares de dichas caras en nuestro editor de UVs. De esta forma vamos 'desenrollando' la piel de nuestro personaje para poder texturarla posteriormente. El proceso de mapeado de una malla es tedioso. En nuestra aplicación, la malla tendrá un mapeado por defecto, que es el que se usará como coordenadas de mapeo. Este mapeado ha sido obtenido mediante un proceso manual a priori, y asegura la adaptabilidad de las diferentes texturas a la mayoría de las mallas que la aplicación sea capaz de generar.



Ejemplo de proyección planar sobre el torso.

Una vez definido el mapeado del personaje, estamos en disposición de generar las texturas que usará el mismo.

Una textura aplicable a nuestro modelo es un archivo de imagen que alberga aquel mapa de color que queremos aplicar a nuestro personaje. Dicho mapa es pintado en relación con el mapa de UVs del mismo, haciendo coincidir la pintura de las diferentes zonas del cuerpo con las zonas en donde están desplegadas sus respectivos mapas de UVs. Así, como podemos apreciar en la imagen superior, la musculatura y detalles del torso correspondiente a una capa de textura de tipo piel roja están ubicados en la zona correspondiente al mapeado de UVs de la zona del torso.

Para aquellas texturas que lo requieran, se confeccionará a si mismo un mapa de opacidad de la misma, también llamado canal alpha. Esta textura tiene por fin determinar qué zonas de la textura de color serán transparentes a la hora de ser aplicadas sobre el modelo y qué zonas serán visibles. Así mismo, el canal alpha puede permitir que determinadas zonas de textura gocen de cierta transparencia.

Un canal alpha es, en síntesis, una imagen en escala de grises. Las zonas de la textura que correspondan a partes negras de su canal alpha asociado serán invisibles en el modelo. Análogamente, las partes de la textura que correspondan a zonas blancas en su canal alpha asociado se verán totalmente una vez aplicadas al modelo. Cualquier valor intermedio hará la zona de textura transparente en mayor o menor medida. El uso de canales alpha nos permitirá solapar texturas para añadir detalles y complementos a nuestro personaje.

Las texturas se combinarán a modo de capas, y en el proceso final de generación del personaje se fusionarán en un solo archivo de imagen.

3.2.5.1. Requisitos técnicos de la textura.

Las texturas usadas serán cuadradas, y potencias de 2. Así, podríamos usar dimensiones de 128x128, 256x256, 512x512, 1024x1024, etc. La imposición de que las texturas sean cuadradas viene dada por varios factores: Maya administra más rápido texturas cuadradas potencia de 2, y muchos motores gráficos no aceptan texturas que no cumplan estos estándares (a efectos de tratamiento de las texturas en tarjetas gráficas, que sean cuadradas y potencia de 2 facilita en gran medida el trabajo).

El tamaño a elegir dependerá de la resolución que necesitemos. Si nuestro personaje tuviese unas texturas muy detalladas y sabemos que va a ser enfocado en una aplicación determinada muy de cerca, es posible que necesitásemos texturas del orden de 1024x1024. Para planos muy lejanos y texturas poco detalladas, una resolución de 256x256 nos es más que suficiente. En el caso que nos ocupa, y por ser G.A.M.A. una aplicación de propósito general, buscaremos una solución de compromiso que pasa por usar un tamaño de textura de 512x512.

Tanto las texturas como sus respectivos canales alpha tendrán este tamaño.

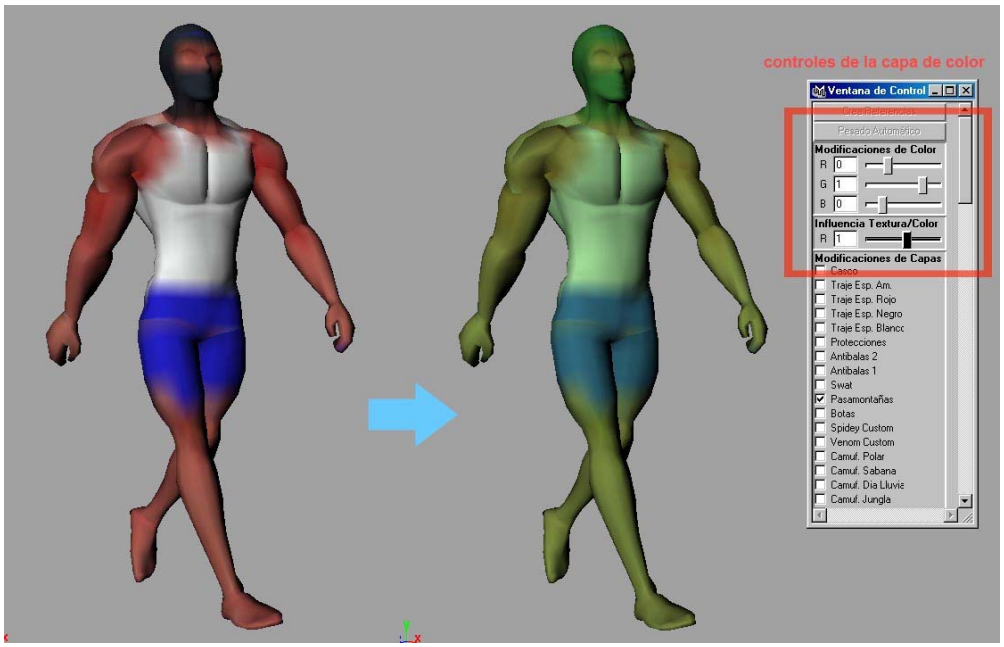
3.2.5.2. Sistema de textura por capas.

La interfaz para generar las texturas presenta una serie de texturas organizadas de forma jerárquica. El usuario tiene la posibilidad de activar o desactivar los complementos de tipo textura que considere necesarios, mostrándose estos en el visor o en las ventanas de render. Si el usuario activa dos elementos de textura que son excluyentes – dos tipos de piel distintos, por ejemplo- se nos mostrará el que se encuentre más arriba en la jerarquía.

Aquellas texturas que representan complementos en la vestimenta o accesorios varios son texturas que llevan asociadas un canal de transparencia, de forma que al ser activadas dejan visibles las texturas que estén por debajo de ella en las zonas en la que la textura no tenga influencia.

De esta forma se consigue estructurar un sistema de texturas que permite configurar la verstimenta y fisonomía de un personaje, disponiendo de un amplio juego de combinaciones posibles.

Por encima de todas las texturas, contamos con una capa de color. La capa de color es una textura dinámica configurable por el usuario. Esta textura es lisa y cubre al resto de texturas sin excepción. Podemos configurar el porcentaje en el que ésta capa de color se funde con el resto de las texturas, pudiendo así desactivarla por completo.



Ejemplo de uso de la capa de color.

La capa de color es configurable, disponiendo asi mismo de tres controladores para los canales RGB y un controlador con el que disponemos en qué porcentaje se fusiona dicha capa con el resto de la textura.

Cuando hemos configurado la textura correctamente, disponemos de la posibilidad de generar un archivo de imagen que fusione todas las capas de textura visibles, condición necesaria para poder exportar dicha textura a cualquier motor gráfico.

Las texturas resultantes serán asi mismo de un tamaño de 512x512 pixels en formato TGA.

3.2.5.3. Jerarquía de nodos en el sistema de texturas por capas en Maya.

La gestión de las texturas se realizará de forma transparente al usuario. En la escena base que gestiona la aplicación, existe un nodo principal de tipo *layeredTexture* que tiene la propiedad de almacenar múltiples capas de textura. El nodo *layeredTexture* gestiona internamente las diferentes capas de textura como una lista de elementos, siendo accesible cada una de las capas mediante programación, llamando directamente a una capa determinada del nodo. Cada capa de textura ofrece diferentes posibilidades. Entre ellas cabe destacar:

- Opciones de fusión: determina en qué forma se fusiona la capa actual con las restantes capas, tanto las superiores como las inferiores.
- Mapa de color: permite asociar un archivo de imagen a una determinada capa.
- Mapa de transparencia: permite asociar un determinado canal alpha a cada una de los mapas de color.

Si una capa no tiene asociado ningún mapa de transparencia, el nodo *layeredTexture* considera que toda la imagen ha de ser mostrada, como si su mapa de transparencia fuese entero blanco. La asignación de capas de color y capas de transparencia al nodo *layeredTexture* se ha realizado de forma automática a nivel de programación, si bien esta opción no está accesible al usuario para modificar el nombre y número de capas presentes en la escena base.

Dependiendo del índice que ostente cada capa de textura en el nodo *layeredTexture*, así vendrá dada su prioridad con respecto a las restantes capas de textura. En concreto, a mayor índice en el nodo, mayor prioridad a la hora de ser mostrado en la textura del personaje.

Paralelamente, existe un nodo de tipo *rampTexture* que es el encargado de gestionar la capa de color aplicable al personaje. A este nodo accederemos mediante programación, accediendo a los canales RGB que controlan el color del nodo.

Ambos nodos, el *rampTexture* y el *layeredTexture*, están conectados a un nodo de tipo *blendColors*. Un nodo de tipo *blendColors* tiene dos canales de entrada, en los cuales se puede asociar dos tipos de texturas diferentes. El primer nodo conectado, el nodo *layeredTexture*, aportará al nodo *blendColors* la información sobre las capas de textura seleccionadas en cada momento. Así mismo, el nodo *rampTexture* aportará la información referente al color seleccionado para la capa de textura que tiene por misión teñir de forma homogenea toda la textura. El nodo *blendColors* administra internamente el par de texturas de entrada, fundiendo en un porcentaje determinado las dos y devolviendo la textura resultante de esta combinación en su parámetro de salida. El parámetro que determina en qué medida se combinan ambas texturas es controlado por programación por el usuario mediante el deslizador de fusión de la capa de color con la textura.

La salida del nodo *blendColors* es la textura que conectaremos a la entrada del material aplicado a la malla. Por defecto, la malla tendrá aplicado un material de tipo *Lambert*, en cuyo canal de color estará conectado la salida del nodo *blendColors*.

La posibilidad de añadir más nodos a esta estructura de árbol que administra las texturas es directa, y permite una gran libertad a la hora de combinar y modificar texturas, siendo posible fusionar más nodos de tipo *layeredTexture* o bien texturas de tipo procedural que incorpora Maya por defecto.

Animación del modelo.

La animación de un modelo no es más que la información sobre la rotación y translación de la jerarquía de huesos asociada al personaje a lo largo del tiempo.

Los recursos de animación que usa la aplicación provienen de dos fuentes bien diferenciadas:

- Escenas de Maya animadas de forma manual por un animador tradicional.
- Archivos de captura de movimiento en formato .bvh, que contienen la infomación sobre la animación de jerarquía de huesos.

A partir de cualquiera de las dos fuentes, necesitamos crear lo que en Maya se denomina un 'clip de animación', esto es, un archivo que contiene información precisa sobre la animación correspondiente a una determinada jerarquía de huesos. Esta información, encapsulada en un archivo de formato binario, es importada mediante programación a petición del usuario, y transferida su información al personaje que estemos creando. De esta forma, podemos importar aquellas animaciones que nos sean de utilidad sin necesidad de que ellas estén presentes al inicio de la creación de nuestro personaje.

En el caso de que la animación venga dada de manos de un animador que ha generado la misma usando las herramientas de animación propias de Maya, necesitaremos seguir los siguientes pasos para convertir dicha animación en un clip de animación válido para ser importado por la aplicación:

1. Crear un *character* de nuestra jerarquía de huesos. Un *character* es una lista con los nombres de los huesos que usamos, y los canales que queremos importar o exportar del mismo –rotación para todos los huesos y además translación para el hueso raíz-.

2. Dar *keys* a todos los huesos en todos los instantes del tiempo. Por defecto, cada canal de animación de un hueso –esto es, las propiedades del mismo, vease rotación o en caso de la raíz, rotación y translación- viene con varias *keys* a lo largo del tiempo, pero no en todos los *frames* o instantes del tiempo. Una *key* o clave de animación es un concepto que emana de la animación tradicional, que corresponde a lo que antes era un fotograma de referencia que se usaba para hacer un previo de una animación de dibujos animados. En un programa de 3D, una *key* equivale a fijar una cierta información de rotación y translación sobre un hueso en un instante del tiempo. El software 3D analiza todas las *keys* presentes en una animación, y para los instantes de tiempo en los que no exista *key*, realiza una interpolación de las mismas, generando lo que se denomina curvas de animación. Para realizar una correcta exportación de la animación, deberemos fijar las rotaciones y translaciones de todos los huesos en todos los instantes de tiempo en los que transcurra la animación. Para ello usaremos por programación la operación *bake channels* que incorpora Maya.

3. Exportar como 'clip de animación' las rotaciones y translaciones de todos los huesos que estén reflejados en el *character* de la jerarquía de huesos. De esta forma, ya tenemos un clip de animación en forma de

archivo binario que contiene la información de animación de nuestro personaje.

En el caso de que la información sobre la animación venga dada como un archivo de captura de movimiento en formato .bvh, habremos de transferir dicha información a una escena de Maya para aplicar posteriormente el proceso antes señalado. Maya no incorpora ninguna herramienta que permita hacer esta labor de forma automática. Para tal fin, hemos usado un script de libre distribución. Existe software propietario de Autodesk Alias, como Motionbuilder, que permite realizar tal función.

Todos los clips de animación generados serán almacenados en formato Maya Ascii e importados mediante programación por la aplicación a petición del usuario.

Generación de recursos de animación.

Existen dos maneras muy diferenciadas de conseguir recursos de animación para gráficos 3D.

El primero de ellos es la **Animación Tradicional**, que tiene sus orígenes en los años 30, y fue perfeccionándose impulsada por las mejoras tecnológicas. Se pasó de primitivas animaciones creadas con fines propagandísticos, a la creación de los primeros largometrajes (Disney, años 50), que han ido evolucionando con el tiempo hasta la actualidad, en la que los ordenadores son un recurso imprescindible. En la actualidad existen grandes empresas dedicadas al género de la animación tradicional, tanto 2D (estudios Gibli) como 3D (Disney's Pixar), que explotan al máximo las últimas tecnologías para ganar en calidad gráfica.

Fotograma de Pinocchio, uno de los primeros largometrajes de animación de la historia.

En esencia, la animación tradicional consiste en crear la ilusión de movimiento dibujando un personaje en distintas poses. Inicialmente esto se hacía dibujando a mano frame por frame (un frame es cada una de los fotogramas estáticos que

se aprecia cuando vemos una animación. En cine, se reproducen 24 frames por segundo). Se establecían primero unas *poses clave* que definían la esencia de la animación, y los huecos entre ellas se rellenaban con imágenes de intercalado, que establecían la progresión suave de una a otra pose clave.

Hoy en día, la animación 2D también se ve asistida por las últimas tecnologías, y los animadores se ayudan de programas de edición de animaciones para completar sus trabajos.

Parecía intuitivo que se diera el paso hacia la animación 3D, y en esencia la teoría aplicada a este tipo de animación es la misma. Las únicas diferencias con la animación 2D son éstas:

- Es necesaria la creación de un esqueleto, que será el que soporte los cambios del personaje y los transmita a su malla.
- En el mundo de las 3D, desaparece la generación manual del intercalado entre poses clave, ya que ahora se utiliza la interpolación automática (es el ordenador quien se encarga de establecer las transiciones de una pose a otra, y después sólo hay que hacer retoques).
- La complejidad de los recursos ha aumentado mucho. Se ha pasado de usar un lápiz sobre un montón de papeles, a utilizar complejos programas de edición de animaciones.
- Es difícil alcanzar el dinamismo de la animación 2D. Cuando se anima sobre 2D, el personaje es completamente modificable, ya que se redibuja con cada nuevo frame. En 3D, sin embargo, el personaje es un objeto existente en la escena, y por tanto está limitado (existen límites en los ángulos de flexión o escalado del personaje, etc. Hoy en día, las grandes empresas como Pixar empiezan a aplicar tecnologías punteras para conseguir los mismos efectos que se usas con facilidad en las 2D).

El segundo método de animación 3D es la **Animación por Captura de Movimientos**. En esta rama se utilizan complejos sistemas de sensores, que aplicados al objeto que ha de simularse (normalmente seres humanos), captan su movimiento y lo almacenan. Estos datos son trasladados después a un esqueleto 3D de fisionomía similar a la de la fuente de la animación. Así, somos capaces de imitar de manera fiel el movimiento de un objeto o ser animado real, y aplicárselo a un objeto o personaje tridimensional.

Andy Serkis, actor que da vida a Gollum en "El Señor de los Anillos", mediante el uso de animación por captura de movimientos.

Los campos de aplicación de este segundo método suelen ser algunos videojuegos (normalmente juegos realistas, como de futbol, tenis, etc.), y el mundo del cine (efectos especiales).

Como puede apreciarse, el criterio en que basarse para decidir si utilizar un método u otro, suele ser si se buscan resultados realistas en detrimento de obtener animaciones más espectaculares o fantásticas, o viceversa.
La mayoría de los videojuegos utilizan animación tradicional, dado que normalmente se espera encontrar dinamismo de acción, y movimientos espectaculares.

Arquitectura de la aplicación y tecnología.

3.3.1. Software de programación utilizado

El sistema de programación utilizado para la creación de G.A.M.A es MEL.

MEL (Maya Embedded Languaje) es un lenguaje de script que funciona sobre Maya, y permite controlar todas las funciones de esta aplicación mediante comandos de programación.

En realidad, cualquier funcionalidad ejecutada con Maya se traduce en la ejecución de un comando MEL relacionado con ésta.

MEL es un lenguaje de programación imperativa, y posee todas las características típicas de cualquiera de estos lenguajes (uso de procedimientos, funciones, bucles, etc.).

Además, MEL ofrece la posibilidad de crear muy fácilmente interfaces visuales, como ventanas con botones, sliders, combobox, etc.

De esta manera, la utilización de MEL es una buena opción ante la necesidad de trabajar con elementos 3D, ya que a pesar de ser bastante sencillo e intuitivo, posee todos los recursos de programación necesarios para hacer herramientas complejas y eficientes, y permite utilizar el amplio abanico de operaciones que nos ofrece Maya de Autodesk Alias.

3.3.2 Tipos de datos en MEL

MEL trabaja con los siguientes tipos de datos:

```
boolean
integer
scalar
vector
transform
color
shader
```

```
lightprofile
string
struct { }
array type
```

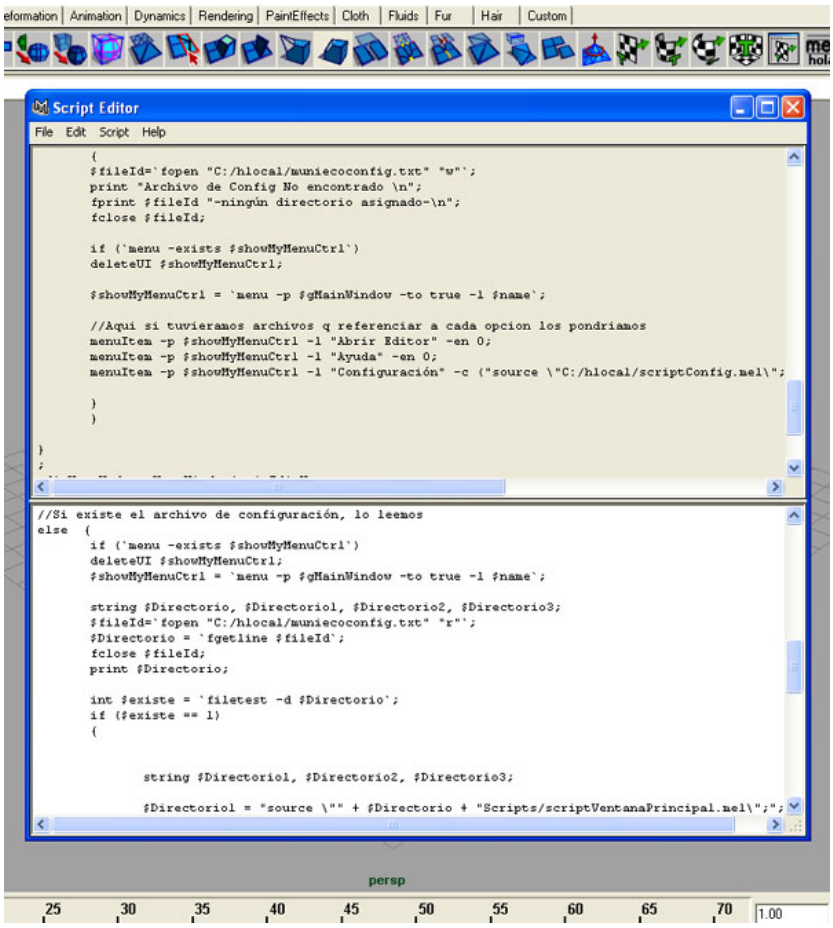
Como se puede observar, entre los tipos se encuentran los más comunes en los lenguajes operativos, además de otros necesarios para trabajar con datos y funcionalidades de Maya.

MEL no trabaja con punteros, y las estructuras han de implementarse con tipos de datos estáticos, como arrays o structs.

3.3.3 Método de trabajo: Programación y depuración.

Los archivos de script MEL son archivos de texto con extensión **.mel**, no encriptados. Por esto mismo, pueden ser editados perfectamente desde el block de notas, o cualquier otro editor de texto predeterminado.

Para depurar el código escrito, es necesario abrir el **Script Editor** dentro de Maya, y cargar allí el programa deseado.



Captura del script editor.

MEL no separa el proceso de depuración entre fase de compilación y fase de ejecución. Sólo existe fase de ejecución. Por esto, el método de depuración es el siguiente: Se programa, y cuando se termina se ejecuta. Si existe algún error en el programa, éste aparecerá en la ventana de status del Script éditor de Maya, y el programa parará la ejecución.

Este proceso puede hacerse algo lento y poco intuitivo, dado que es necesario ejecutar cada vez que se desea depurar, y a veces el origen del error no aparece correctamente especificado en la barra de status.

3.3.4 Llamadas a funciones y procedimientos

Como hemos dicho antes, en MEL pueden definirse llamadas a funciones y procedimientos. Esto facilita mucho la estructuración y permite claridad en el código.

Todas las funciones y procedimientos han de declararse en la parte superior del programa. En la parte inferior quedará la zona de ejecución, donde se puede llamar a las funciones y procedimientos definidos anteriormente.

Existe la posibilidad de declarar variables globales, permitiendo así transladar información entre distintos procedimientos.

3.3.5 Estructura general de una aplicación: Llamada a otros scripts.

En MEL está permitido llamar a un programa de script desde el programa actual. De esta manera, mediante el comando **source,** paramos la ejecución del programa actual, y ejecutamos por completo el programa del archivo seleccionado, para continuar con la ejecución de nuestro código después.

No está permitido el intercambio de información entre dos tramos de código MEL implementados en archivos diferentes. Por tanto, la comunicación entre programas distintos habrá de realizarse mediante el uso de **ficheros.**

La posibilidad de estructurar un programa en distintos tramos de código que se llaman unos a otros es muy adecuada a la hora de desarrollar programas complejos y extensos. Esta estructura se asemeja de alguna forma a las estructuras de clases que pueden crearse en programación orientada a objetos. Se obtienen así características importantes, como transparencia, existencia de niveles de abstracción, y facilidad de ampliación o modificacion.

En nuestro código, hemos estructurado nuestro programa encapsulando de forma intuitiva los tramos de código basándonos en las diferencias de sus funcionalidades, o su nivel de abstracción.

Con este método de estructuración pueden desarrollarse fácilmente diagramas o mapas que describen de manera intuitiva la topología del programa (como veremos en el apartado siguiente).

3.3.6 Estructura general del programa

Procederemos primero a definir la estructura general de nuestro programa, resaltando su topología, e indicando los distintos módulos que componen la totalidad de la aplicación

3.3.6.1 Diagrama de clases

Esquema inicial

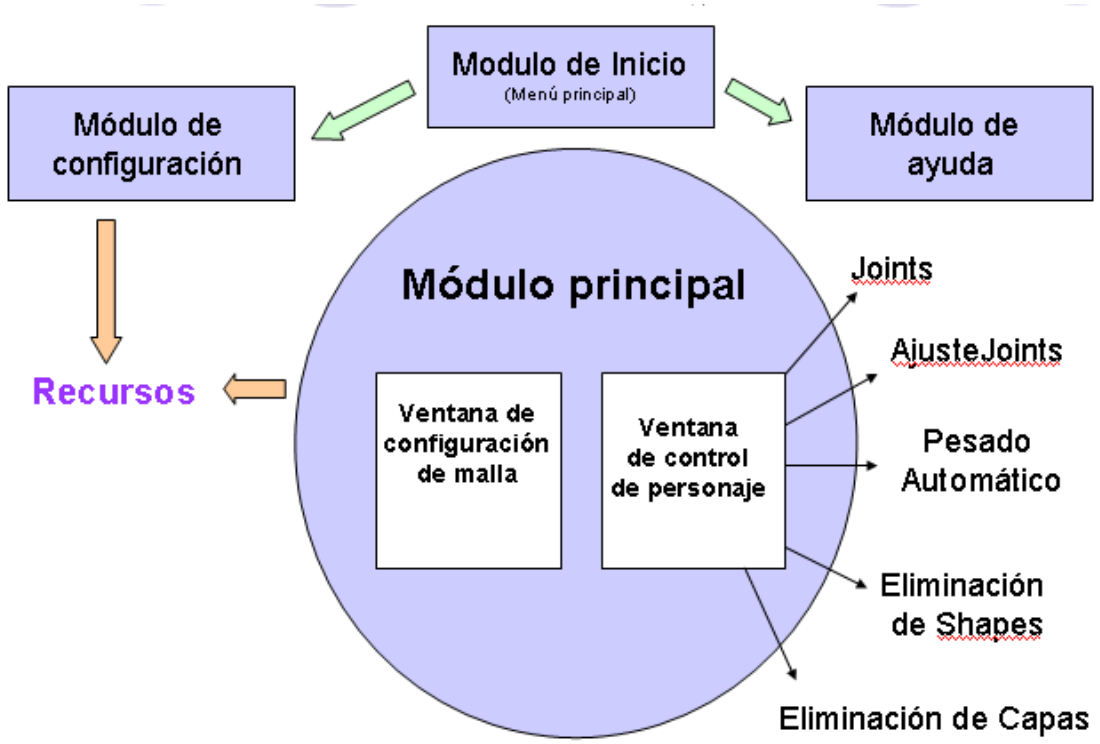


Diagrama de la arquitectura de la aplicación.

El archivo de script que ha de cargarse inicialmente es **ScriptMenuPrincipal**. Este programa se encarga de hacer aparecer un nuevo menú en la barra de menús de Maya, llamado **G.A.M.A**. A través de el podemos acceder a tres nuevas ventanas: La ventana de configuración, la de ayuda, y la Ventana Principal (cada una de las tres implementadas como programas de script diferentes).

Ventana principal se encarga de poner en pantalla las dos ventanas necesarias para la creación de un personaje: Ventana de customizado, y ventana de control.

La **ventana de control** posee varios botones que llaman al resto de archivos de script auxiliares, que realizan operaciones necesarias para crear la estructura final, necesaria para exportar

3.3.6.2 Descripción detallada de cada uno de los módulos:

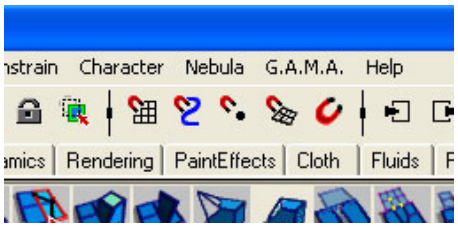
Menú Principal (scriptMenuPrincipal.mdl)

Como se explicó antes, es el archivo que hay que cargar para hacer funcionar el programa.

Es el fragmento de código de más alto nivel.

Posee las siguientes funcionalidades:

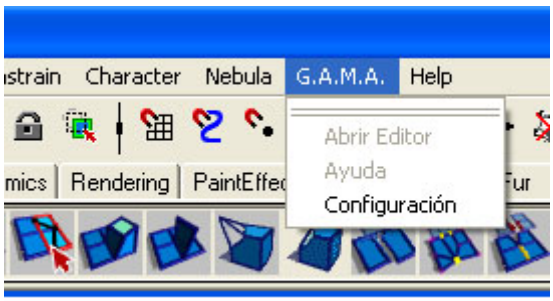
- Mostrar el menú en la barra de menús de maya



Generamos un menú propio de la aplicación.

- Desplegar las tres opciones principales: Configuración, Ayuda, y Abrir el Editor de Personajes

- Gestionar la configuración del plugin. Si no encuentra el archivo de configuración en el directorio por defecto, significa que aún no se ha configurado el programa. Por tanto, se encarga de bloquear las otras dos opciones. Así, el usuario se ve obligado a pinchar en la única opción disponible, la de configuración. Una vez que se ha configurado, las otras dos opciones se habilitan, permitiendo al usuario comenzar a crear el personaje, o leer la ayuda.



Screenshot de la opción configuración del menú.

El menú creado por este script permanece siempre en pantalla, permitiendo al usuario reconfigurar y ver la ayuda en todo momento, o bien recargar el generador de personajes sin causar ningún conflicto.

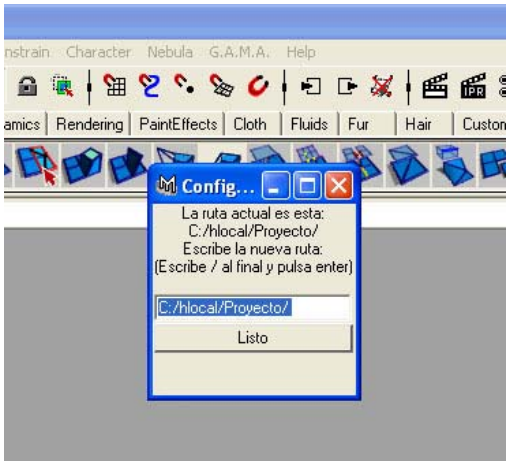
La implementación de este script no conlleva excesivos problemas, debido a que básicamente se trata de manejar UI-Elements (elementos de interfaz con el usuario), teniendo en cuenta todos los posibles casos de uso que pueden darse.

`Ventana de Ayuda (scriptAyuda.mel)`

Primera opción dentro del menú principal. Se encarga de crear una ventana con un texto de ayuda.
La ventana es configurable.
El script se encarga de comprobar antes si ya existe una ventana de ayuda creada. Si es así, elimina ésta, para evitar repeticiones.

`Ventana de Configuración (scriptConfig.mel)`

Script encargado de crear una ventana para la configuración del path (directorio) de trabajo del plug-in.



Screenshot sobre la opción de configuración.

En la ventana aparece un campo de texto que se encarga de recibir el path indicado por el usuario. Cuando el usuario pulsa la tecla “enter” aparece un botón en la ventana, para confirmar los datos introducidos. Cuando se pulsa este botón, la ventana desaparece.

Funcionamiento de la configuración del directorio de trabajo:

El directorio de trabajo del plugin se almacena en un archivo de texto llamado “muniecoconfig.txt”. En este archivo aparece, en código ascii, la ruta que hay que seguir para encontrar la carpeta “Proyecto” de nuestro plugin.

El funcionamiento es el siguiente:
En la fase inicial (antes de haber configurado por primera vez) no existe el archivo “muniecoconfig.txt”. Cuando el menú principal (definido anteriormente) detecta que no existe tal archivo, lo crea, y deja escrita en él la siguiente

cadena: "-ningún directorio asignado-“. Cuando el usuario abre la ventana de configuración (única opción que le aparece disponible), se muestra por pantalla que no existe ninguna ruta de acceso asignada. De esta manera, el usuario escribe la ruta (siguiendo las normas indicadas más adelante), y ésta es guardada en el archivo de texto citado anteriormente.
Así, a partir de entonces el programa se encarga de ir a buscar todos los recursos necesarios (otros archivos de script, mallas de personaje, mapas de pesado, texturas, etc.) a la ruta indicada por el archivo de texto.

El path escrito por el usuario ha de seguir las siguientes normas tipográficas para ser correctamente identificado por el programa:

- Comienzo de la unidad en mayúscula (ej. C:)
- Las barras de separación de las carpetas han de ser /, y no \.
- Hay que escribir una barra al final del path.
- Existe distinción entre mayúsculas y minúsculas.
- El usuario ha de pulsar enter al final de la cadena

En caso de existir ya una ruta definida, cuando el usuario vuelve a configurar la ruta, la antigua es sustituida por la nueva.

Este programa también comprueba que no exista una ventana con el mismo nombre, para evitar repeticiones.

Para realizar esta funcionalidad del programa, ha sido necesario investigar el manejo de ficheros de texto con MEL.

`Ventana Principal (scriptVentanaPrincipal.mel)`

Este archivo de script engloba todas las funcionalidades del editor de personajes.
Al ejecutarse, abre dos ventanas diferentes, que nos permitirán realizar todos los pasos necesarios para crear desde cero un personaje animado.

Las dos ventanas editables se crean a la vez, y son las siguientes:

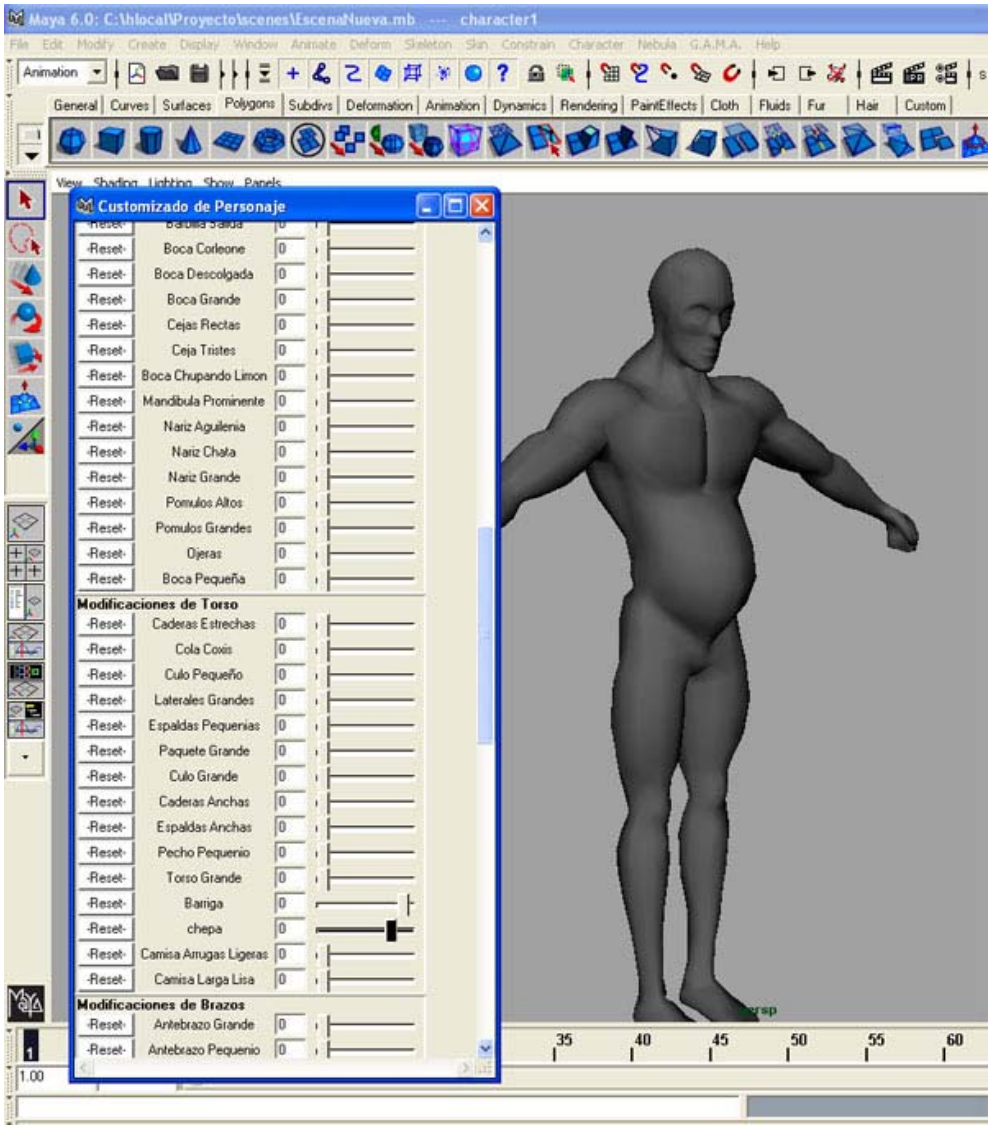
Ventana de Configuración de malla:

Es la ventana que utilizaremos durante la primera fase de creación del personaje.
Con ella iremos definiendo la malla poligonal del personaje, variando sliders (deslizadores) que actúan directamente sobre la geometría del personaje.
Estos sliders están conectados por programación con las diversas shapes creadas específicamente para hacer variaciones del personaje. Como se ha explicado en apartados anteriores, todas las variaciones son combinables entre sí, de manera que se establece una interpolación coherente de la malla, haciendo infinitas las posibilidades.

Cada uno de los deslizadores va acompañado de un campo de texto donde puede escribirse un valor para el deslizador a mano, y un botón de reset, que coloca el slider en su posición original.

Los deslizadores están agrupados por las siguientes zonas corporales:

- Cabeza y cuello
- Cara
- Torso
- Brazos
- Piernas
- Modificaciones especiales
-



Ventana de configuración de la malla desplegada.

En la parte inferior de esta ventana, aparecen tres nuevas funcionalidades:

- Botón “Generar Back-up Malla”

Se encarga de ejecutar un comando que guarda las modificaciones de la maya generadas hasta el momento.

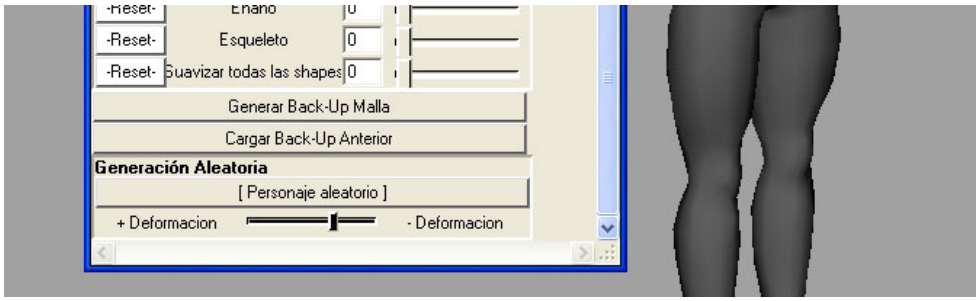
Lo que se hace realmente es guardar un archivo llamado backupmalla.ma en la carpeta del proyecto.
Decidimos implementar esta funcionalidad para dar opción al usuario de guardar el personaje generado sin terminar el proceso de creación, y poder seguir con el resto de los pasos en otro momento.
Sólo se permite guardar un archivo de back-up, y si generamos uno nuevo se advertirá que el anterior back-up será sobreescrito.
Después de haber pulsado el botón puede seguirse modificando la malla del personaje, de manera que el usuario puede ir haciendo back-ups de seguridad sin necesidad de preocuparse.

- Botón “Cargar Back-up Anterior”:

Llama a un procedimiento interno llamado “importamalla”, que se encarga de eliminar la malla actual e importar la guardada anteriormente. La ejecución de este botón implica el cierre de la ventana de customizado, por lo que el usuario se ve forzado a seguir con los pasos lógicos de creación, usando la única ventana activa, la ventana de control.

- Grupo “Generación Aleatoria”

Está compuesto por un botón llamado “[Personaje Aleatorio]”, y un deslizador. Cada vez que el usuario pulsa el botón, se genera una deformación aleatoria del personale.
El slider influye en la proporción de deformaciones aleatorias que se ejercen de una vez sobre el personaje. Si colocamos el slider a la derecha, al pulsar el botón no se obtendrá ningún efecto. Si lo colocamos a la izquierda, absolutamente todas las deformaciónes serán aplicadas en menor o mayor medida a la malla del personaje.
El usuario podrá ir graduando con posiciones intermedias del slider la cantidad de deformación que quiere para su personaje aleatorio.
El funcionamiento interno de la generación aleatoria es el siguiente:
Cada vez que se pulsa el botón de generación aleatoria, se crea por cada deformación un valor random (de 0 a 1, establecido por la función interna de MEL rand()), que determina si esa deformación en particular se aplicará al personaje (establecemos que si el valor ofrecido por el random es mayor que el valor obtenido del slider, se aplica la deformación. En caso contrario, esa deformación no se aplica). En caso positivo, lo hará de nuevo con una influencia de otro valor de 0 a 1 establecido con otra función random.
Cabe mencionar que para asegurar que el random no llega a repetir valores, cambiamos para cada ejecución del boton la semilla generadora.



Vista inferior de la ventana de configuración de la malla.

Una vez finalizado el proceso de configuración del personaje malla, el usuario seguirá el proceso natural de uso, ejecutando el primer botón de la otra ventana, (Ventana de Control), y éste cerrará la Ventana de Configuración de malla.

En la parte final del código de esta ventana se encuentra el sistema de protección de la escena.
Este sistema consiste en abrir la escena base original, y hacer una copia al instante. De esta forma, cerramos el original y abrimos la copia, de manera que el archivo original nunca puede ser modificado.

Tecnología para la interpolación de modificaciones:

Para conseguir la producción de deformaciones en la malla poligonal de un personaje poseemos una malla base, y tantas otras mallas como deformaciones existan. Cada una de estas mallas de deformación poseen una estructura idéntica a la de la base, y sólo difieren por tanto en las posiciones de los vértices.

El tipo de interpolación que queremos producir se denota en infografía como *paralelo*, y consiste en lo siguiente:
Cada una de las deformaciones influirá sobre la base de forma paralela. Los vértices que se mantienen en la misma posición que en la malla base no influirán en las deformaciones, y para el resto se aplicará una media de la posición de la base y la de todas las deformaciones.

Una de las partes de la representación de una malla poligonal en Maya es la posición de cada uno de los vértices que la componen (se dice una de las partes debido a que una malla poligonal viene definida por otros muchos parámetros, como la composición de caras, la dirección de las normales de cada una de estas caras, su material, etc.).
La posición de cada uno de estos vértices se nos ofrece en Maya como un array de vectores (un vector es un array de tres reales, que indican las posiciones X,Y y Z de cada vértice).

Para acceder a los vértices, se sigue la siguiente notación:

NombreDeLaMalla.vtx[nº de vértice]

Con nº de vértice que va desde 0, a tantos vértices como tenga la maya menos uno.

Bloque de pseudocódigo
Definimos una malla como un array de vértices
Vd[ ] es una malla de deformación
Vm[ ] es la malla original
Vf[ ] es la malla final
Cte[ ] es un array con números que van de 0 a 1, y definen la influencia de cada una de las deformaciones sobre la maya original.

```
Para n=1 hasta N (con N = nº de vértices de la malla)
   Posición = Vm[n]
      Para m=1 hasta M (con M= nº de deformaciones existentes) hacer
         Si Vd[n]  (para la malla de deformación m) /= Vm[n] entonces
            Contador = Contador + 1
            Posición = Posición +  (Cte[m] * Vd[n])
      FPara
      Vf[n] = Posición/Contador
      Contador = 1
FPara
```

Básicamente, lo que hacemos es establecer una media aritmética de las posiciones de todos los vértices que difieren en posición con los vértices de la malla original.

Vamos seleccionando vértice por vértice, y viendo en cuales de las deformaciones difieren las posiciones para ese vértice. Tenemos en cuenta sólo estas posiciones para establecer la media aritmética que define la interpolación para ese vértice.

Es necesario hacer notar que a la hora de establecer la media aritmética, también se tiene en cuenta una constante, que indica en qué grado influye la deformación seleccionada, y que es determinada por cada uno de los sliders de las deformaciones.
El algoritmo termina cuando se recorren todos los vértices de la malla.

En el pseudocódigo no tenemos en cuenta que las posiciones son vectores de 3 elementos (X,Y,Z).

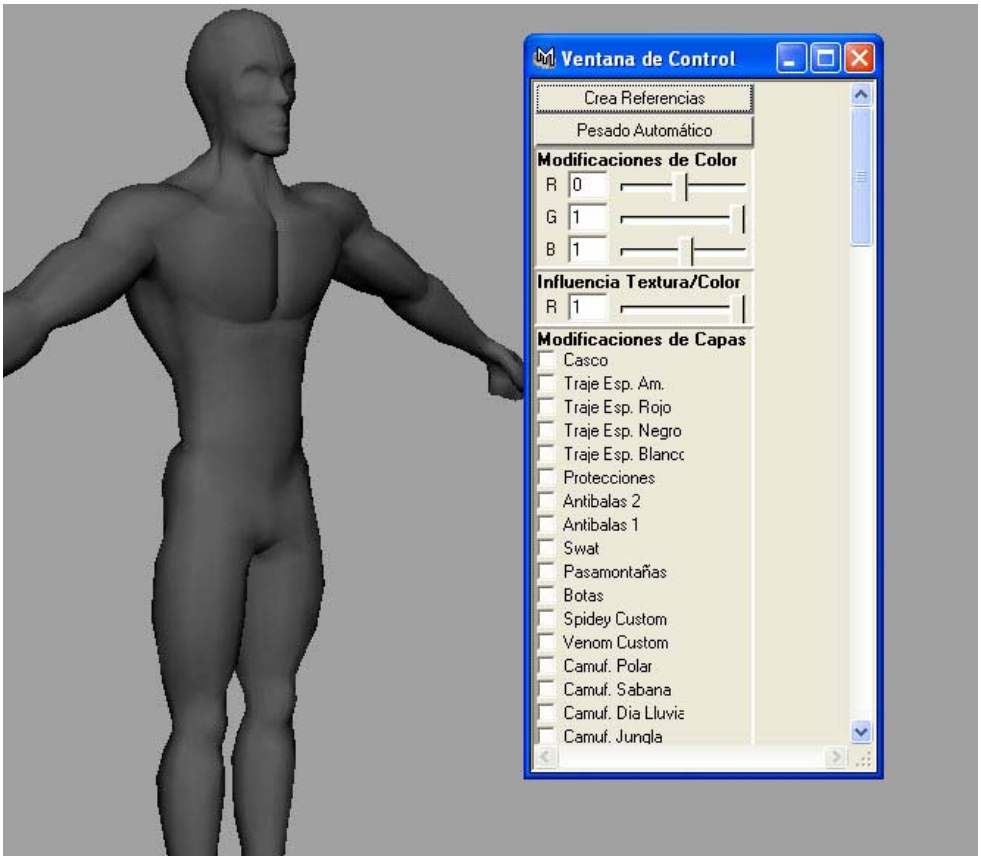
En realidad, cada una de las deformaciones se guarda en un vector de deformaciones.

Ventana de Control

Esta ventana pasará a usarse después de haber terminado la configuración de la malla, y contiene las funcionalidades necesarias para terminar de preparar el personaje para exportar. Los botones están diseñados esperando que el

usuario siga un orden secuencial, de arriba a abajo. Para ello, establecemos todos los bloqueos necesarios entre botones.

Dada la complejidad de las operaciones implementadas dentro de cada botón, existen llamadas a scripts auxiliares, que aquí se nombran, y después pasarán a ser definidos en profundidad.



Vista de la ventana de control de la aplicación.

Iremos describiendo las distintas fases del proceso de creación con esta ventana, junto con los elementos de la ventana que las implementan:

Botón “Crea referencias”
Al pulsar este boton se realizan las siguientes funciones:

- Se crean las referencias para los huesos, llamando al script auxiliar scriptJoints.mel
- Se eliminan las shapes de la escena (llamando al script auxiliar, scriptBorraShapes.mel).
- Se cierra la ventana de configuración de malla para evitar que el usuario siga modificando la malla.
- Se desactiva este botón (el usuario no puede volver a pulsarlo una vez utilizado).

NOTA: Se asume que entre la ejecución del botón y la del siguiente, el usuario habrá desplazado a su gusto las referencias para el esqueleto si lo desea.

Botón "Pesado Automático"

Funciones realizadas al pulsar este botón:

- Se llama al script auxiliar scriptAjustahuesos.mel, que se encarga de colocar el verdadero esqueleto allá donde estén las referencias.
- Se llama al script auxiliar scriptPesadoAutomático.mel, que pesa la malla al esqueleto.
- Se desactiva este botón, para que no vuelva a ser usado.
- Se importa un mapa de pesado creado anteriormente, que mejora el comportamiento del pesado automático realizado anteriormente.

Tecnología del pesado de una malla

Pesar una malla poligonal significa atribuir a cada uno de sus vértices un grado de influencia de alguno de los huesos del esqueleto.

El tipo de pesado utilizado para nuestro proyecto es *pesado suave.* Esto significa que se permite la influencia de distintos huesos del esqueleto para un mismo vértice.

En esencia, el proceso realizado para conseguir que la malla poligonal se mueva en relación con el esqueleto es el siguiente:

A cada uno de los vértices se le asignan dos de los huesos del esqueleto, determinando el grado de influencia de cada uno de los dos. Ambos grados de influencia han de sumar uno.

Cuando se importa un clip de animación al esqueleto, se aplican en cada frame de la animación las matrices de rotación, translación y escalado de cada hueso, a dos copias auxiliares de cada uno de los vértices de la malla. De cada una de estas dos copias auxiliares de cada vértice transformado obtenemos un promedio determinado por el grado de influencia del que hablábamos en el párrafo anterior. Este promedio (una posición, al fin y al cabo) es el que aportamos como valor definitivo para cada uno de los vértices de la malla en cada frame de una animación.

$$\text{Posición Vértice} = \frac{[(k1 * \text{AplicaMatricesTransform}(\text{Hueso1})) + (k2 * \text{AplicaMatricesTransform}(\text{Hueso2})]}{2}$$

Con k1 y k2, grados de influencia de los huesos para cada vértice.

Grupo de sliders "Modificadores de color"
Estos tres sliders modifican el color general del personaje. Cada uno de ellos define la influencia de cada uno de los tres colores del formato RGB (rojo, verde, azul). Estos sliders están conectados a cada una de las componentes de la ColorEntryList de la Rampa de color de una textura.

Slider de Influencia Textura/Color
Este slider sirve para definir cuánto influye el color establecido por el grupo de sliders anterior sobre la textura que después crearemos. La capa de color se funde con la capa final de textura mediante un nodo blendColors, que mezcla ambas en una proporción determinada por el parámetro de este slider.
Si el slider se coloca totalmente a la derecha, sólo se verá la textura. Si se coloca totalmente a la izquierda, se verá en su totalidad del color definido anteriormente.

Conjunto de checkbox “Modificaciones de Capas”:
La textura final que tendrá el personaje está formada por la unión de múltiples capas de textura. Esta unión se consigue mediante el uso de un nodo LayeredTexture, que contiene todas las capas de textura junto con sus mapas alpha. Los mapas alpha de cada capa definen qué parte se verá y qué parte será transparente en cada capa, en caso de ser esta activada.
La unión de todas las capas no es una fusión en este caso, si no una superposición. De esta manera, las capas superiores se verán siempre ocultando a las inferiores si están activadas.
Cada uno de los combobox está asignado a cada una de las capas, y en caso de marcarse, activan la visibilidad de su capa.
De esta manera, el usuario irá definiendo la textura final que quiere para su personaje.

Tecnología de mezcla de capas

El formato de las imágenes utilizadas para crear la textura es un RGB multicapa. Después de establecer la unión de todas las capas mediante un nodo LayeredTexture, hemos de determinar qué parte de cada una de las capas visibles se verá en la imagen final.

Para ello nos valemos de otra imagen multicapa que posee un mapa de transparencia análogo a cada una de las capas anteriores. Hemos establecido que los mapas de transparencia no tengan degradado, facilitando así las cosas.

Por tanto, un mapa de transparencia es una imagen que contiene pixels blancos o negros. Para cada una de las capas de textura, mostramos sólo los pixels cuyo anexo en el mapa de transparencia sea blanco.

Una imagen no codificada viene representada normalmente por una matriz de pixels, teniendo en cuenta que cada píxel es una terna de valores que representan la cantidad de Rojo, Verde o Azul de ese píxel.

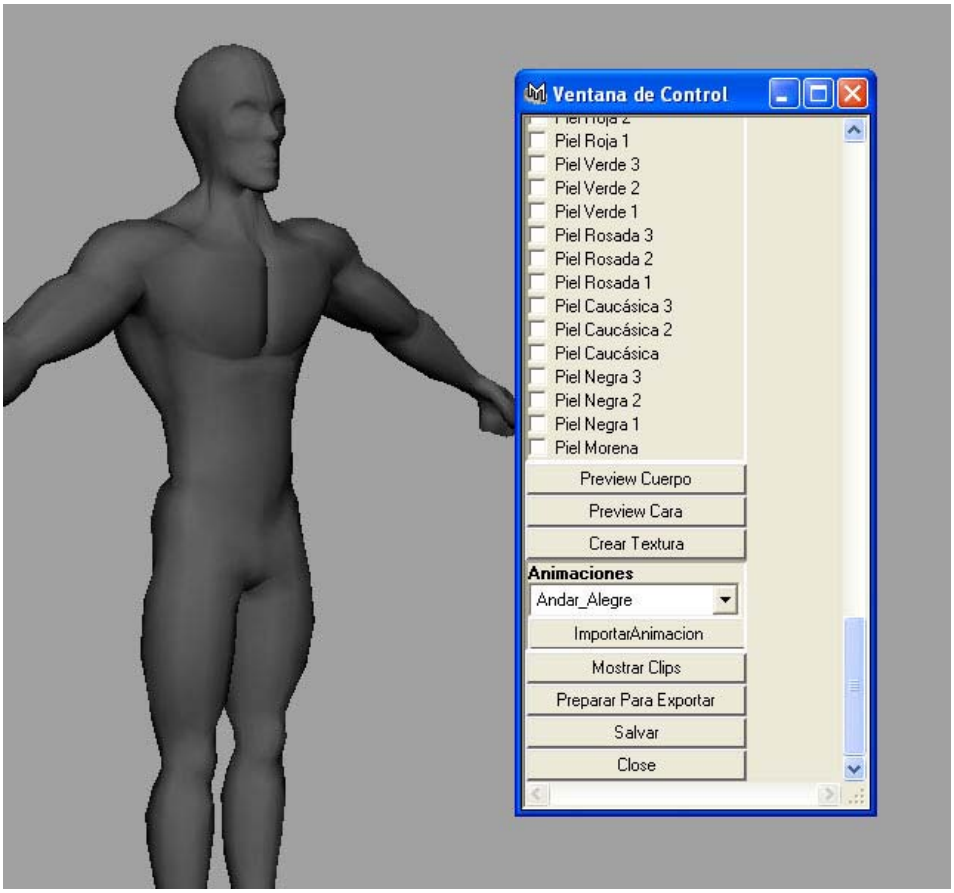
Así, iremos recorriendo todas las capas de texturas, desde la menos prioritaria hasta la de mayor prioridad. Para cada uno de sus pixels, copiamos a la textura final su color sólo si el píxel de su mapa de transparencia es blanco.

Nótese que en el proceso descrito anteriormente pueden sobrescribirse valores de color de un píxel, si una capa de nivel superior también tiene en blanco el mismo píxel de su mapa de transparencia.

Bloque de Pseudocódigo

Representamos una imagen como un array de pixels, y un píxel como un array de enteros, que van de 0 a 255.
Tenemos dos arrays que contienen cada una de las capas y cada uno de los mapas de transparencia (ArrayDeCapasTransparencia[ ] y ArrayDeCapas[ ]).

```
Para m=0 hasta nº de capas, hacer
   CapaActualTransparencia = ArrayDeCapasTransparencia[m]
   CapaActual = ArrayDeCapas[m]
   Para n=0 hasta 512*512 hacer (todos los pixels de una imagen)
      Si CapaActualTransparencia[n]=(255,255,255) hacer
         ImagenFinal[n]=CapaActual[n]
      Fsi
   FPara
Fpara
```



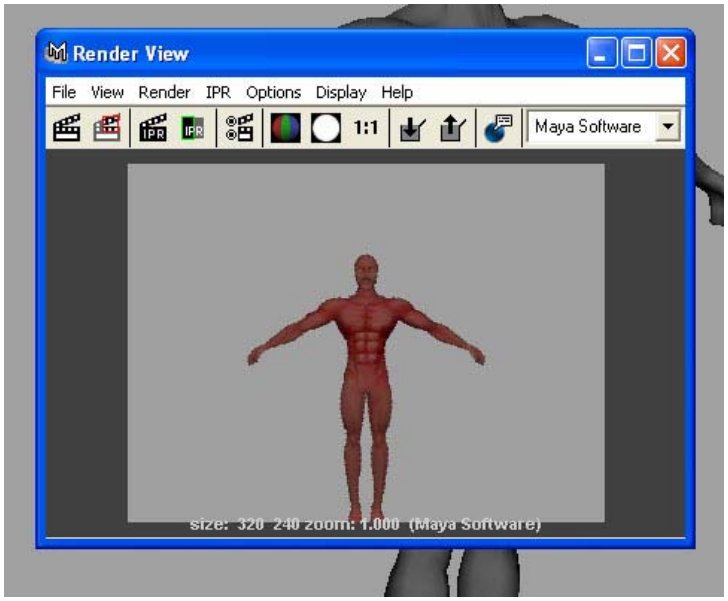
Vista de la parte inferior de la ventana de control de la aplicación.

Botón "Preview Cuerpo"

Al pulsar este botón se llama a un procedimiento auxiliar llamado *rendear*, que se encarga de abrir una ventana de *Render View*, y crear un snapshot de cuerpo entero del personaje, a una resolución de 320x240 pixels.

De esta manera, el usuario puede comprobar cómo va quedando la textura que genera con la funcionalidad anterior.

Este preview puede realizarse en cualquier paso de la creación del personaje, y está preparado para seguirlo sin perder el enfoque en caso de que ya haya sido animado. Para conseguir esto último, huvo que crear una nueva cámara, y hacerla hija del hueso de mayor nivel en la jerarquía del esqueleto, de manera que los movimientos del esqueleto son adoptados también por esta cámara.



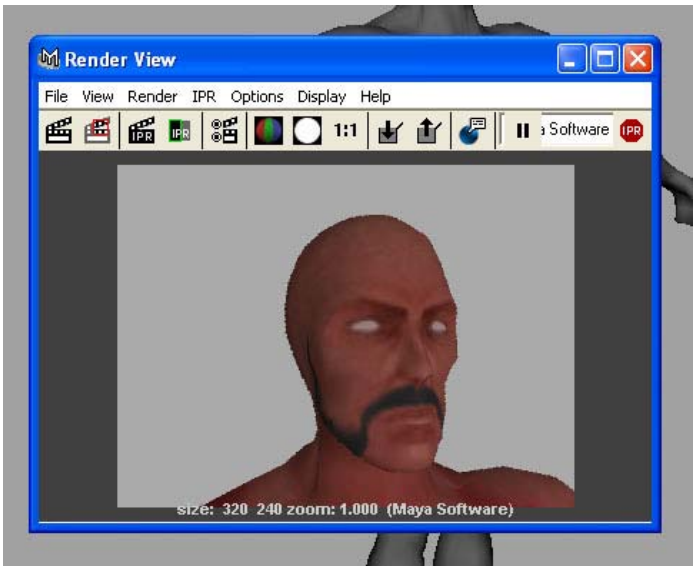
Ventana desplegada por la opcion 'preview cuerpo'

Botón "Preview Cara"

Este botón tiene la misma funcionalidad que el anterior, con la diferencia de enfocar en este caso la cara del personaje.

Para mantener el enfoque en este caso, se creó una nueva camara y se le hizo hija en jerarquía del hueso de la base del cuello del personaje.

El botón llama a otro prodecimiento auxiliar llamado *rendear2* que realiza las operaciones descritas.



Ventana desplegada por la opción 'preview cara'

Botón "Crear Textura"

Al pulsar este botón se llama a un comando de Maya que se encarga de compilar todas las capas de textura seleccionadas por el usuario y dejar el resultado en una imagen monocapa con formato tga, y resolución de 512x512

De esta manera se genera el archivo de textura que el usuario utilizará al realizar la exportación del personaje.

Esta imagen se crea en el directorio de trabajo del plugin.

Esta función conlleva una carga importante de procesamiento, dado que hay que integrar un número elevado de capas con transparencias en una sola imagen. El tiempo de generación de la textura varía entre uno o dos minutos, dependiendo del ordenador que se utilice.

Campo "Animaciones"

Está compuesto de un combobox y un botón.

El combobox contiene los nombres de todas las animaciones disponibles por el usuario para añadir a su personaje.

El botón "Importar Animación" añade la animación seleccionada actualmente en el combobox. Para hacerlo, se llama a un procedimiento auxiliar llamado "importaclip".

El proceso real de este botón es la importación de un clip de animación con el nombre seleccionado en el combobox, en un *Character Set* creado específicamente para el esqueleto que vamos a utilizar.

Los clips se encuentran ya almacenados en la carpeta Clips del proyecto.

Al final de la importación, se coloca por comando el cursor del *timeline* en el frame 1 de la animación. De esta manera, cuando el usuario pulse el botón de *play* la animación se verá desde el principio.

Botón "Mostrar Clips"

Este botón se encarga de llamar por comando a un editor de clips de animación. De esta manera, cuando se pulsa se abre una nueva ventana que contiene un bloque por cada animación importada. El usuario puede seleccionar (pinchando con un click izquierdo de ratón) cada uno de los bloques, y arrastrarlo en el *timeline, o bien suprimirlo (pulsando la tecla supr).*

Mediante la combinación de el campo "Animaciones" y este botón, el usuario puede ir previsualizando todas las animaciones, y quedarse al final sólo con las que le gustan.

Botón "Preparar para exportar"

Este botón se encarga de preparar la escena tal y como la requerirá el exportador de Nebula2.

Para ello, se realizan los siguientes pasos:

- Se desconecta el nodo BlendColors
- Se eliminan todas las capas de textura de la escena (Para esto se llama a un archivo de script auxiliar llamado scriptBorraCapas.mel).
- Se agrupa la geometría en un nodo llamado "model", tal y como exige el exportador de Nebula2.
- Se desactivan los botones de preview.

Botón "Salvar"

Este botón abre una ventana standard de "Salvar cómo".
El directorio inicial por defecto es el directorio del proyecto, pero el usuario puede cambiarlo si lo desea.

El formato en que se salva es el .mb (Maya Binary), formato Standard para las escenas de Maya.

De esta manera, el usuario puede guardar la escena antes de exportar a Nebula2, pudiendo dejar para otro momento este último proceso.

Botón "Close"

Cierra la ventana.

Script para el ajuste del esqueleto ("scriptAjustaHuesos.mel")

Este script se encarga de mover todos los huesos que componen el esqueleto real a la posición que ocupan las referencias.
Básicamente lo que se hace es coger el atributo translate de las referencias (que son locators), y aplicárselo a cada uno de los huesos.

Como se dijo antes, este script es invocado cuando se pulsa el botón “Pesado Automático”.

Script para el borrado de capas (“scriptBorraCapas.mel”)

Este script borra una por una todas las capas existentes en la escena inicial. Se le llama al pulsar el botón “Preparar Escena”.

Script para el borrado se shapes (“scriptBorraShapes.mel”)

Este script elimina una por una todas las shapes pertenecientes a cada una de las deformaciones posibles de la malla. Este script es invocado cuando se pulsa el botón “Crea Referencias”, momento en el cual se sabe que no se van a modificar las deformaciones.

Script de ajuste automático de referencias (“scriptJoints.mel”)

Este script auxiliar se encarga de colocar las referencias en lugares estratégicos de la malla, ajustando automáticamente y de una manera bastante fiel las referencias a donde deberían ir, para que el pesado de la malla funcione correctamente.

Tecnología para colocación automática de centros de giro.

Básicamente se trata de establecer un centro de giro coherente para cada uno de los huesos, de manera que al girar estos, los vértices de la malla se comporten de una manera fisionómicamente correcta.

El procedimiento utilizado es el siguiente. Se cogen las posiciones de dos vértices relevantes de la zona de la malla que debería ocupar cada uno de los huesos, y la posición que toma la referencia es una media de la posición de los dos vértices.

Por ejemplo, para colocar el centro de giro del hueso denominado “muñeca”, es intuitivo seleccionar dos vértices enfrentados de la zona de la muñeca, y hacer su media.

Entero a = determinaVertice1(“Muñeca”);

Entero b = determinaVertice2(“Muñeca”);
PosiciónHuesoMuñeca[0] = (Dummy.vtx[a][0]+Dummy.vtx[b][0])/2
PosiciónHuesoMuñeca[1] = (Dummy.vtx[a][1]+Dummy.vtx[b][1])/2
PosiciónHuesoMuñeca[2] = (Dummy.vtx[a][2]+Dummy.vtx[b][2])/2

Con *determinaVertice1(string), determinaVertice2(string)* funciones que devuelven el índice real de los dos vértices determinantes de la malla para un hueso determinado

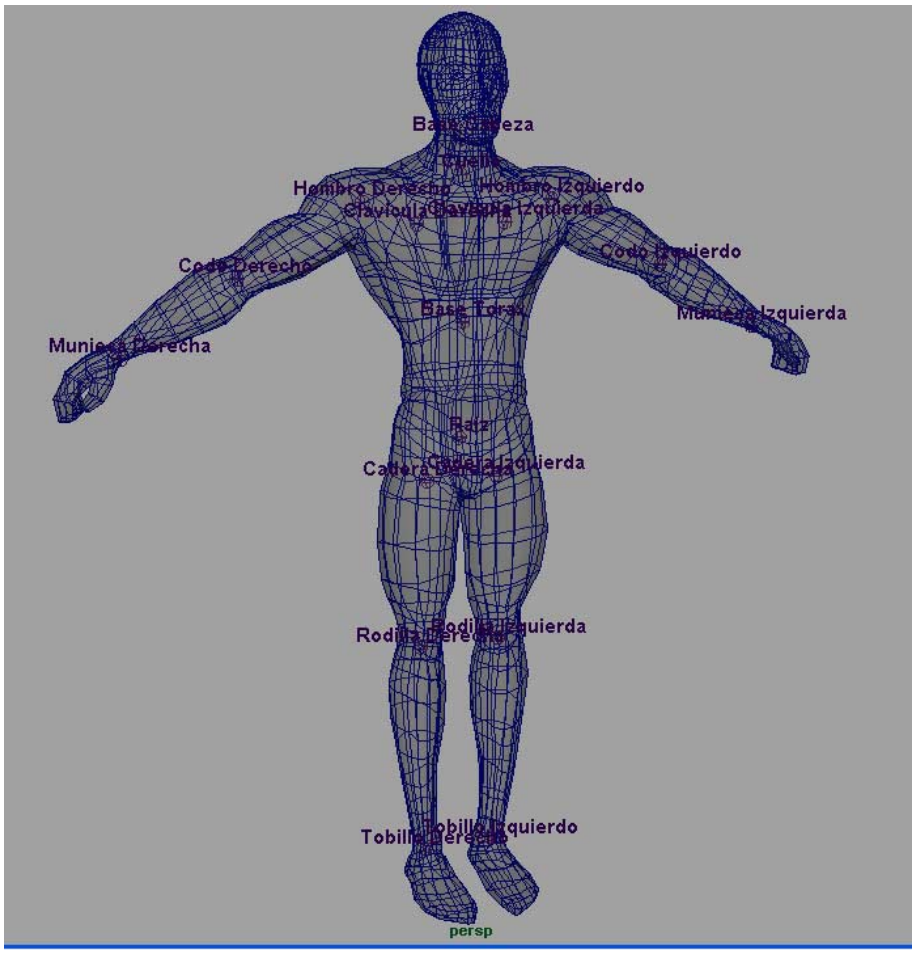
Los valores 0, 1 y 2, como índices de los vectores indican cada uno de los valores de las tres componentes de un vértice en tres dimensiones.

Básicamente se trata de establecer un centro de giro coherente para cada uno de los huesos, de manera que al girar estos, los vértices de la malla se comporten de una manera fisionómicamente correcta.
El procedimiento utilizado es el siguiente. Se cogen las posiciones de dos vértices de la malla de la zona que debería ocupar cada uno de los huesos, y la posición que toma la referencia es una media de la posición de los dos vértices.

Por ejemplo, para colocar el centro de giro del hueso denominado “muñeca”, es intuitivo seleccionar dos vértices enfrentados de la zona de la muñeca, y hacer su media.

De esta manera, las referencias se encontrarán colocadas de entrada en un lugar bastante aceptable, y rara vez tendrá el usuario experimentado que recolocar alguna de las referencias para ajustar su centro de giro.

Además de esta operación, en este script se cambia el modo de vista del visor a *“Wireframe + X–Ray”* (Modo de rayos X con distinción de aristas), y se hacen visibles las referencias, para que el usuario pueda moverlas fácilmente si lo desea.



Ejemplo del proceso de colocación automática de los vértices.

Script de pesado automático de la malla (“scriptPesadoAutomático.mel”)

En este script se realiza por comando el pesado automático de la malla, estableciendo los pesos de cada vértice por proximidad a los huesos más cercanos.

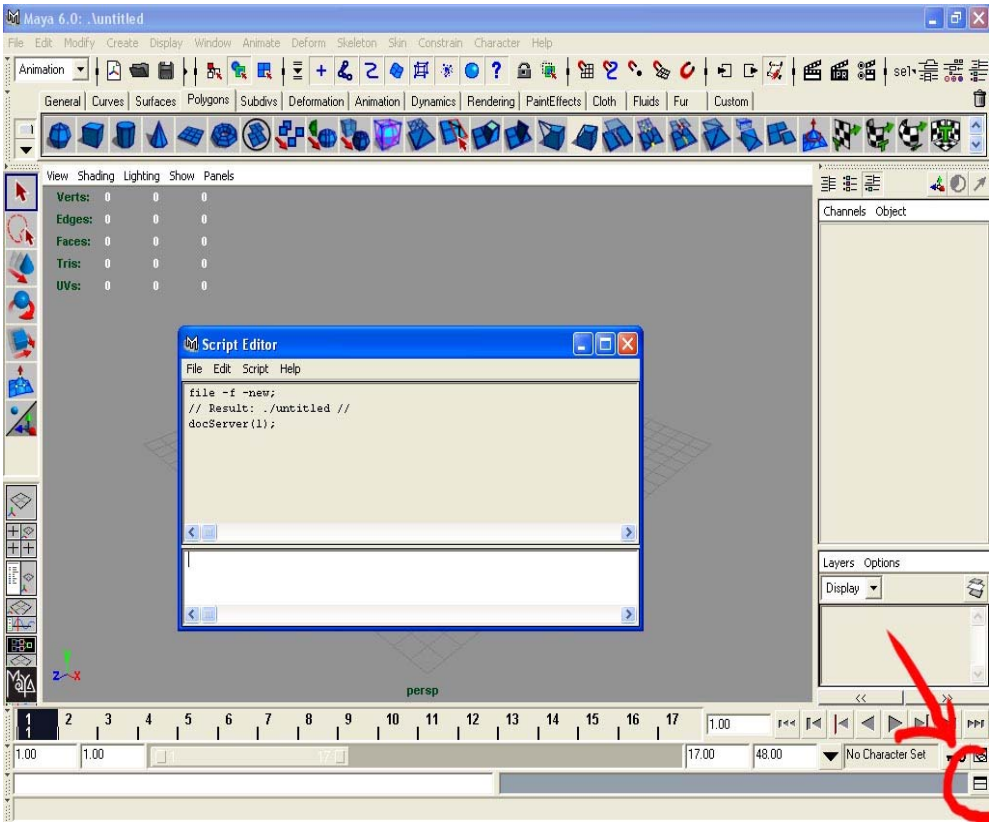
El tipo de pesado es un *smooth bind*, con influencia de dos huesos por cada vértice.

Además, este script cambia el modo de vista a su estado original, y elimina las referencias, puesto que ya han cumplido su función.

Flujo de trabajo de la aplicación.

La secuencia de uso del programa, desde la fase de instalación del mismo, hasta la finalización de un personaje listo para exportar, es la siguiente:

Fase de Instalación:
1. Copiar los archivos "ScriptMenuPrincipal.mel" y "ScriptConfig.mel" en la carpeta C:/hlocal/.
2. Copiar la carpeta Proyecto donde esté guardado el Plugin**.**
3. Abrir Maya.
4. Abrir el Script Editor



Ubicación de la pestaña para abrir el script editor.

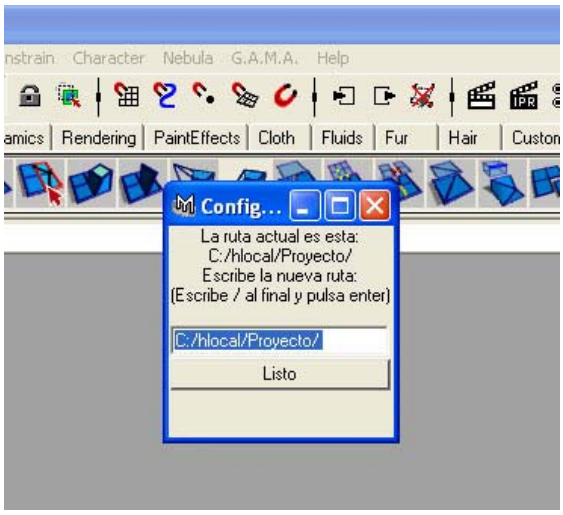
5. En la ventana del Script Editor, abrir el script “ScriptMenuPrincipal.mel” (File → Open Script).
6. Ejecutar el Script (Script → Execute). Aparecerá un nuevo menú en la ventana principal de Maya.

Fase de Configuración:
En esta etapa configuraremos el path del directorio de trabajo del proyecto, para poder acceder a los recursos necesarios del programa.

Al abrir la ventana de configuración se nos pide el path del directorio de trabajo. Es necesario seguir las siguientes normas tipográficas para introducirlo de manera correcta:

- Comienzo de la unidad en mayúscula (ej. C:)

- Las barras de separación de las carpetas han de ser /, y no \.
- Hay que escribir una barra al final del path.
- Existe distinción entre mayúsculas y minúsculas.
- El usuario ha de pulsar enter al final de la cadena.



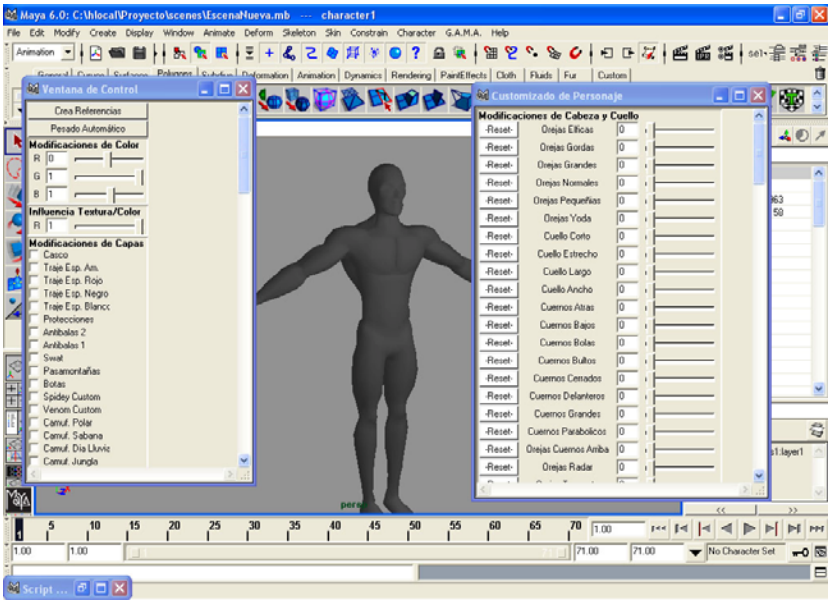
Ejemplo de configuración.

Para guardar los cambios realizados, habrá de pulsarse el botón “Listo”.

Después de realizar por primera vez la fase de configuración, se activan las otras dos opciones del menú G.A.M.A.

Fase de Trabajo

2. Accedemos a la opción “Abrir Editor”. El plugin quedará listo para ser usado.

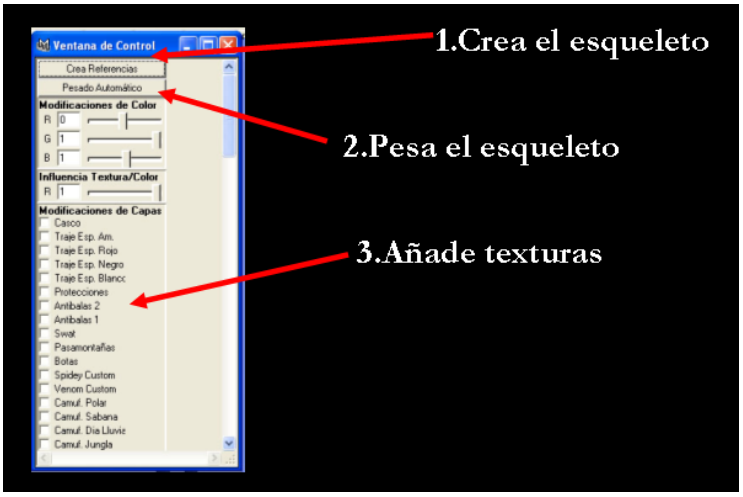


Captura que muestra el aspecto general de nuestra aplicación.

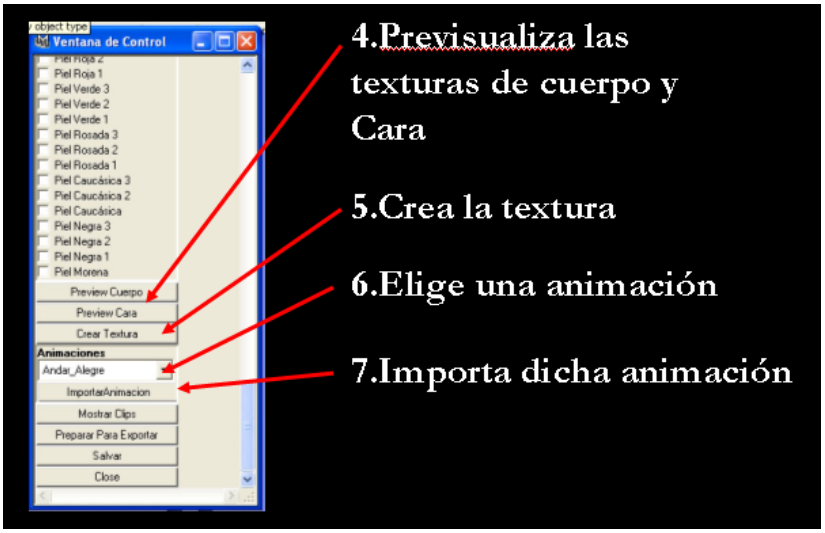
3. Etapa de Configuración de la Malla:
    - En la ventana de Configuración de la Malla, variamos todos los sliders necesarios para obtener un modelo a nuestro gusto.
    - Podremos utilizar, en caso de ser necesario, las opciones de guardado y carga de un backup de la malla.
    - También podremos valernos del apartado de generación automática de un personajes, para generar personajes aleatorios, y utilizarlos directamente, o variar sus características hasta obtener el modelo deseado.

4. Etapa de Pesado, Texturizado, y Animación del Modelo:

    - Cambiamos de ventana, y pulsamos el botón “Crea referencias”. Se generan así las referencias auxiliares para creación y ajuste del esqueleto.
    - Pulsamos el botón de “Pesado Automático”. El programa aprovecha el posicionamiento establecido de las referencias, para realizar el pesado automático de la malla al esqueleto real.
    - Establecemos las modificaciones deseadas de color y capas de textura.
    - Realizamos las previsualizaciones necesarias de las texturas, con los botones de “Preview”.
    - Pulsando el botón “Crear Textura” generamos la textura monocapa que utilizaremos como textura final del personaje en el motor gráfico.
    - Elegimos las animaciones deseadas, y las previsualizamos valiéndonos del interfaz de reproducción de maya, y de los botones “Importar Animación” y “Mostrar Clips”.
    - Usamos el botón “Preparar Para Exportar” para hacer las modificaciones necesarias en la escena, de manera que el toolkit de exportación para el motor genere de manera correcta el archivo para el personaje.
    - Guardamos la escena con el botón “Salvar”, por si queremos realizar la exportación en otro momento.



Parte superior de la ventana de control.



Parte inferior de la ventana de control.

Hasta aquí llega la secuencia de uso natural para nuestro proyecto. Cabe resaltar que inmediatamente después de la misma, existiría una última fase de exportación final del personaje, que se realiza con un sencillo software externo al proyecto, e interpreta los datos generados por G.A.M.A. para adaptarlos al sistema de datos del motor gráfico.

4. Experiencias de uso, conclusiones y trabajo futuro.

4.1. Experiencias de uso.

G.A.M.A. ha sido testada y utilizada en proyectos reales relacionados con el mundo de los videojuegos y el entretenimiento en el Master en Desarrollo de Videojuegos impartido en la Facultad de Informática de la Universidad Complutense de Madrid, en el periodo comprendido entre Junio y Julio de 2006.

Tras una sesiones introductorias sobre el manejo y finalidad de la aplicación, se distribuyeron diferentes copias de la aplicación a los integrantes del Máster. A través de diversos formularios sobre el funcionamiento de la aplicación y recogiendo impresiones por parte de los usuarios, se afinaron aquellos aspectos que los usuarios hicieron notar como mejorables, a saber: conjunto de animaciones y deformaciones, distribución de la interfaz de usuario y funcionalidad del módulo de set-up.

Realizadas las mejoras antes mencionadas, se obtuvo una versión final suficientemente intuitiva y estable para ser usada en entornos de desarrollo de aplicaciones 3D de forma rápida, simple y eficiente.

4.2. Conlcusiones.

Mediante el uso de nuestra aplicación, es posible sustituír las labores de búsqueda de recursos gráficos, así como las de adaptación de los recursos al motor gráfico utilizado.

Así mismo, contratar a un grafista para realizar las labores requeridas por un proyecto de desarrollo gráfico, conlleva una importante inversión económica, que puede suplirse por el hipotético precio de nuestra herramienta, que se amortizaría rápidamente, en caso de utilizarse para varios proyectos.

Si bien esta aplicación no es capaz de realizar gráficos tan detallados y específicos como los obtenidos del trabajo de un grupo de grafistas, si es capaz de generar prototipos de personajes que superan con creces en el apartado artístico lo que se espera de un personaje para un prototipo.

Por todo esto, G.A.M.A. se perfila como una aplicación ideal para el prototipado rápido y efectivo de videojuegos y el desarrollo íntegro de los recursos 3D referentes a personajes necesarios para un proyecto de investigación de ámbito universitario.

4.3. Trabajo futuro.

El diseño de la aplicación GAMA permite una extensión de la misma de una forma rápida y secuencial. A priori, podemos distinguir dos tipos bien diferenciados de ampliaciones: ampliaciones directas en la propia aplicación y evolución de la aplicación.

4.3.1 .Ampliaciones directas y evoluciones de la aplicación..

Con el término de ampliación directa nos referimos a aquellas mejoras que se podrían implementar con un mínimo coste y que mejorarían la funcionalidad de la aplicación, pero que no representarían una evolución en las prestaciones de la misma. Ejemplo de este tipo de ampliaciones serían añadir nuevas deformaciones a la malla, nuevos tipos de textura, nuevas animaciones, etc. Estas ampliaciones no suponen mucho esfuerzo, pues se basan en la modificación de la escena base en torno a la que funciona la aplicación –a la hora de incorporar, por ejemplo, nuevas deformaciones- y reflejar esos cambios en los correspondientes módulos de script. La facilidad de ampliación en este sentido radica en el hecho de que ningún elemento básico en el funcionamiento de la aplicación está ‘cerrado’ y son, por tanto, susceptibles a cualquier tipo de cambio. Asi pues, cambiar el juego de texturas, incluir nuevas texturas, añadir animaciones en forma de clips y demas ampliaciones que pudiesen aportar mayor funcionalidad y potencia a la herramienta son fáciles de implementar en la medida que se disponga de los recursos necesarios. Tales modificaciones se realizarian actualizando paralelamente la escena base y los correspondientes módulos de script.

Como evolución de la aplicación entendemos una ampliación de la misma en la que se incorporen nuevas herramientas que no existían en la anterior. Este tipo

de ampliaciones pueden requerir de la modificación de la escena base y la ampliación o modificación de los diferentes módulos de script.

4.3.2. Posibles ampliaciones.

Ampliaciones directas sobre la aplicación:

- Incorporación de nuevos paquetes de texturas, aumentando así la versatilidad en la producción de personajes.
- Incorporación de nuevos juegos de deformaciones, permitiendo la generación de más tipos de personajes.
- Ampliación del número de animaciones disponibles.
- Remodelación de la interfaz de usuario haciéndola más intuitiva si cabe a ojos del usuario.
- Posibilidad de configuración de la escena para poder exportar el personaje a diferentes motores. En la actualidad, la escena queda preparada para ser exportada al motor gráfico Nebula2.
- Posibilidad de elección del idioma en el que se nos presenta la interfaz de usuario.

Ampliaciones posibles en futuras evoluciones de la aplicación:

- Posibilidad de configurar diferentes tipos de personajes de concepciones muy distintas a elección del usuario. Podríamos elegir entre generar un personaje de estilo cartoon, un personaje femenino, un cuadrúpedo, monstruos…
- Posibilidad de seleccionar las texturas y animaciones que el usuario desee que gestione la aplicación, de forma dinámica y a través de menús amigables.
- Herramienta para la configuración de las animaciones importadas. Con ella podríamos controlar parámetros tales como la velocidad, balanceo y grado de rotación de ciertos huesos para configurar a medida los diferentes juegos de animaciones.
- Selector de canales de bump y mapas de desplazamiento.
- Generador de complementos tales como objetos, armas, etc. configurables de forma dinámica –siguiendo el mismo procedimiento que la configuración de los personajes-.
- Módulo de generación de terrenos y elementos fijos de escenarios.
- Mayor versatilidad en las opciones de salvaguarda de personajes y animaciones.
- Biblioteca de plantilla de personajes.
- Posibilidad de reducir de forma dinámica el número de polígonos del personaje sin perjuicio del resto de prestaciones de la aplicación.

5. Bibliografía.

- **Complete Maya Programming: An Extensive Guide to MEL and C++ API**. Autor: *David Gould*. Editorial Morgan Kaufmann, 2002.

- **Complete Maya Programming, Vol. II: An In-Depth Guide to 3D Fundamentals, Geometry, and Modeling**. Autor *David Gould*. Editorial Morgan Kaufmann, 2005

- **Maya 6: The Complete Referente**. Autor*: Tom Meade, Shinsaku Arima*. Editorial McGraw-Hill Osborne Media; 1 edition , 2004.

- **The MEL Companion: Maya Scripting for 3D Artists**. Autor: *David Stripinis*. Editorial Charles River Media; 1 edition, 2003.

- **3-D Human Modeling and Animation, Second Edition**. Autor: *Peter Ratner*. Editorial Wiley, 2003.

- **The Animator's Survival Kit: A Manual of Methods, Principles, and Formulas for Classical, Computer, Games, Stop Motion, and Internet Animators**. Autor: *Richard Williams*. Editorial Faber & Faber, 2002.

Apéndice A.

Código de la aplicación.

```
// Creation Date: <29-01-06>
//
// ***************************************
//
// Description: Crea un menu en la ventana
// principal de MAYA, llamado G.A.M.A.
//
//***************************************


//Esto crea un menu en la ventana principal de MAYA

global proc myMenu()
{
global string $showMyMenuCtrl;

//Elimina cualquier menu q haya para llegar al menu principal de maya
if (`menu -exists $showMyMenuCtrl`)
   deleteUI $showMyMenuCtrl;

string $name = "G.A.M.A.";
global string $gMainWindow;

$showMyMenuCtrl = `menu -p $gMainWindow -to true -l $name`;

int $existe = `filetest -w "C:/hlocal/muniecoconfig.txt"`;

//Si no existe el archivo de configuración, lo creamos
if ($existe == 0)
    {
    $fileId=`fopen "C:/hlocal/muniecoconfig.txt" "w"`;
    print "Archivo de Config No encontrado \n";
    fprint $fileId "-ningún directorio asignado-\n";
    fclose $fileId;

    if (`menu -exists $showMyMenuCtrl`)
    deleteUI $showMyMenuCtrl;

    $showMyMenuCtrl = `menu -p $gMainWindow -to true -l $name`;

    //Sólo habilitado el boton de configuración
    menuItem -p $showMyMenuCtrl -l "Abrir Editor" -en 0;
    menuItem -p $showMyMenuCtrl -l "Ayuda" -en 0;
    menuItem -p $showMyMenuCtrl -l "Configuración" -c ("source
\"C:/hlocal/scriptConfig.mel\"; ");

    }

//Si existe el archivo de configuración, lo leemos
else  {
    if (`menu -exists $showMyMenuCtrl`)
    deleteUI $showMyMenuCtrl;
    $showMyMenuCtrl = `menu -p $gMainWindow -to true -l $name`;

    string $Directorio, $Directorio1, $Directorio2, $Directorio3;
    $fileId=`fopen "C:/hlocal/muniecoconfig.txt" "r"`;
    $Directorio = `fgetline $fileId`;
    fclose $fileId;
    print $Directorio;

    int $existe = `filetest -d $Directorio`;
    if ($existe == 1)
    {


        string $Directorio1, $Directorio2, $Directorio3;

        $Directorio1 = "source \"" + $Directorio +
"Scripts/scriptVentanaPrincipal.mel\";";
        $Directorio2 = "source \"" + $Directorio + "Scripts/scriptAyuda.mel\";";
```

```
       $Directorio3 = "source \"" + $Directorio + "Scripts/scriptConfig.mel\";";


       menuItem -p $showMyMenuCtrl -l "Abrir Editor" -c ($Directorio1);
       menuItem -p $showMyMenuCtrl -l "Ayuda" -c ($Directorio2);
       menuItem -p $showMyMenuCtrl -l "Configuración" -c ($Directorio3);
   }
   else
   {
   $fileId=`fopen "C:/hlocal/muniecoconfig.txt" "w"`;
   print "Archivo de Config No encontrado \n";
   fprint $fileId "-ningún directorio asignado-\n";
   fclose $fileId;

   if (`menu -exists $showMyMenuCtrl`)
   deleteUI $showMyMenuCtrl;

   $showMyMenuCtrl = `menu -p $gMainWindow -to true -l $name`;

   //Aqui si tuvieramos archivos q referenciar a cada opcion los pondriamos
   menuItem -p $showMyMenuCtrl -l "Abrir Editor" -en 0;
   menuItem -p $showMyMenuCtrl -l "Ayuda" -en 0;
   menuItem -p $showMyMenuCtrl -l "Configuración" -c ("source
\"C:/hlocal/scriptConfig.mel\"; ");

   }
   }

}
;
editMenuUpdate MayaWindow|mainEditMenu;

//Ahora llamamos al procedimiento q acabamos de hacer
myMenu;
setFocus `paneLayout -query -pane1 viewPanes`;
editMenuUpdate MayaWindow|mainEditMenu;
editMenuUpdate MayaWindow|mainEditMenu;
editMenuUpdate MayaWindow|mainEditMenu;


// Fecha: <25-04-06>
//
//*****************************************
//
// Descripcion: Script generador de las ventanas del programa
//
//*****************************************

//----------------------------------------//
//DECLARACION DE VARIABLES Y PROCEDIMIENTOS//
//----------------------------------------//


//Ventana1: Ventana para el customizado del personaje. Contiene sliders que modifican la
geometria del personaje

global proc Ventana1()
{

   // Eliminamos ventanas con el mismo nombre en caso de existir
   if ( `window -exists customizado` ) deleteUI -window customizado;


   string $window = `window -title "Configuración de la malla" -widthHeight 200 55
customizado`;
      string $scrollLayout = `scrollLayout -horizontalScrollBarThickness 16 -
verticalScrollBarThickness   16`; //Crea un scroll
      columnLayout;



   //--------------//
   //SLIDERS CABEZA//
   //--------------//

   frameLayout -label "Modificaciones de Cabeza y Cuello" FrameCambiosCabeza;
```

```
    rowColumnLayout -numberOfColumns 4 -cw 1 50 -cw 2 120 -cw 3 30 -cw 4 100 ;  //Layout
de los deslizadores para las shapes.


       button -label "-Reset-" -command ("setAttr \"orejasElficas1.orejasElficas\"
0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Orejas Elficas" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "orejasElficas1.orejasElficas" );
       connectControl $slider ( "orejasElficas1.orejasElficas" );

       button -label "-Reset-" -command ("setAttr \"orejasGordas1.orejasGordas\" 0.0;")
-backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Orejas Gordas" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "orejasGordas1.orejasGordas" );
       connectControl $slider ( "orejasGordas1.orejasGordas" );

       button -label "-Reset-" -command ("setAttr \"orejasGrandes1.orejasGrandes\"
0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Orejas Grandes" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "orejasGrandes1.orejasGrandes" );
       connectControl $slider ( "orejasGrandes1.orejasGrandes" );

       button -label "-Reset-" -command ("setAttr \"orejasNormales1.orejasNormales\"
0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Orejas Normales" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "orejasNormales1.orejasNormales" );
       connectControl $slider ( "orejasNormales1.orejasNormales" );

       button -label "-Reset-" -command ("setAttr \"orejasPequenias1.orejasPequenias\"
0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Orejas Pequeñias" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "orejasPequenias1.orejasPequenias" );
       connectControl $slider ( "orejasPequenias1.orejasPequenias" );

       button -label "-Reset-" -command ("setAttr \"orejasYoda1.orejasYoda\" 0.0;") -
backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Orejas Yoda" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "orejasYoda1.orejasYoda" );
       connectControl $slider ( "orejasYoda1.orejasYoda" );

       button -label "-Reset-" -command ("setAttr \"cuelloCorto1.cuelloCorto\" 0.0;") -
backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Cuello Corto" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "cuelloCorto1.cuelloCorto" );
       connectControl $slider ( "cuelloCorto1.cuelloCorto" );

       button -label "-Reset-" -command ("setAttr \"cuelloEstrecho1.cuelloEstrecho\"
0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Cuello Estrecho" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "cuelloEstrecho1.cuelloEstrecho" );
       connectControl $slider ( "cuelloEstrecho1.cuelloEstrecho" );
```

```
       button -label "-Reset-" -command ("setAttr \"cuelloLargo1.cuelloLargo\" 0.0;") -
backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Cuello Largo" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "cuelloLargo1.cuelloLargo" );
       connectControl $slider ( "cuelloLargo1.cuelloLargo" );

       button -label "-Reset-" -command ("setAttr \"cuelloAncho1.cuelloAncho\" 0.0;") -
backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Cuello Ancho" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "cuelloAncho1.cuelloAncho" );
       connectControl $slider ( "cuelloAncho1.cuelloAncho" );


       button -label "-Reset-" -command ("setAttr \"cuernosAtras1.cuernosAtras\" 0.0;")
-backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Cuernos Atras" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "cuernosAtras1.cuernosAtras" );
       connectControl $slider ( "cuernosAtras1.cuernosAtras" );

       button -label "-Reset-" -command ("setAttr \"cuernosBajos1.cuernosBajos\" 0.0;")
-backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Cuernos Bajos" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "cuernosBajos1.cuernosBajos" );
       connectControl $slider ( "cuernosBajos1.cuernosBajos" );

       button -label "-Reset-" -command ("setAttr \"cuernosBolas1.cuernosBolas\" 0.0;")
-backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Cuernos Bolas" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "CuernosBolas1.cuernosBolas" );
       connectControl $slider ( "cuernosBolas1.cuernosBolas" );

       button -label "-Reset-" -command ("setAttr \"cuernosBultos1.cuernosBultos\"
0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Cuernos Bultos" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "cuernosBultos1.cuernosBultos" );
       connectControl $slider ( "cuernosBultos1.cuernosBultos" );

       button -label "-Reset-" -command ("setAttr \"cuernosCerrados1.cuernosCerrados\"
0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Cuernos Cerrados" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "cuernosCerrados1.cuernosCerrados" );
       connectControl $slider ( "cuernosCerrados1.cuernosCerrados" );

       button -label "-Reset-" -command ("setAttr
\"cuernosDelanteros1.cuernosDelanteros\" 0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Cuernos Delanteros" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "cuernosDelanteros1.cuernosDelanteros" );
       connectControl $slider ( "cuernosDelanteros1.cuernosDelanteros" );

       button -label "-Reset-" -command ("setAttr \"cuernosGrandes1.cuernosGrandes\"
0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Cuernos Grandes" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
```

```
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "cuernosGrandes1.cuernosGrandes" );
        connectControl $slider ( "cuernosGrandes1.cuernosGrandes" );

        button -label "-Reset-" -command ("setAttr
\"cuernosParabolicos1.cuernosParabolicos\" 0.0;") -backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Cuernos Parabolicos" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "cuernosParabolicos1.cuernosParabolicos" );
        connectControl $slider ( "cuernosParabolicos1.cuernosParabolicos" );

        button -label "-Reset-" -command ("setAttr
\"orejasCuernosArriba1.orejasCuernosArriba\" 0.0;") -backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Orejas Cuernos Arriba" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "orejasCuernosArriba1.orejasCuernosArriba" );
        connectControl $slider ( "orejasCuernosArriba1.orejasCuernosArriba" );

        button -label "-Reset-" -command ("setAttr \"orejasRadar1.orejasRadar\" 0.0;") -
backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Orejas Radar" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "orejasRadar1.orejasRadar" );
        connectControl $slider ( "orejasRadar1.orejasRadar" );

        button -label "-Reset-" -command ("setAttr \"orejasTrompeta1.orejasTrompeta\"
0.0;") -backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Orejas Trompeta" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "orejasTrompeta1.orejasTrompeta" );
        connectControl $slider ( "orejasTrompeta1.orejasTrompeta" );

        button -label "-Reset-" -command ("setAttr \"cabezaGrande1.cabezaGrande\" 0.0;")
-backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Cabeza Grande" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "cabezaGrande1.cabezaGrande" );
        connectControl $slider ( "cabezaGrande1.cabezaGrande" );

        button -label "-Reset-" -command ("setAttr \"cabezaAlargada1.cabezaAlargada\"
0.0;") -backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Cabeza Alargada" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "cabezaAlargada1.cabezaAlargada" );
        connectControl $slider ( "cabezaAlargada1.cabezaAlargada" );

        button -label "-Reset-" -command ("setAttr \"cabezaCuadrada1.cabezaCuadrada\"
0.0;") -backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Cabeza Cuadrada" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "cabezaCuadrada1.cabezaCuadrada" );
        connectControl $slider ( "cabezaCuadrada1.cabezaCuadrada" );

        button -label "-Reset-" -command ("setAttr \"cabezaPequenia1.cabezaPequenia\"
0.0;") -backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Cabeza Pequenia" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "cabezaPequenia1.cabezaPequenia" );
        connectControl $slider ( "cabezaPequenia1.cabezaPequenia" );
```

```
       button -label "-Reset-" -command ("setAttr \"cascoMilitar1.cascoMilitar\" 0.0;")
-backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Casco Militar" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "cascoMilitar1.cascoMilitar" );
       connectControl $slider ( "cascoMilitar1.cascoMilitar" );

       button -label "-Reset-" -command ("setAttr \"crestaPunk1.crestaPunk\" 0.0;") -
backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Cresta Punk" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "crestaPunk1.crestaPunk" );
       connectControl $slider ( "crestaPunk1.crestaPunk" );

       button -label "-Reset-" -command ("setAttr \"cuernosCortados1.cuernosCortados\"
0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Cuernos Cortados" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "cuernosCortados1.cuernosCortados" );
       connectControl $slider ( "cuernosCortados1.cuernosCortados" );

       button -label "-Reset-" -command ("setAttr \"peloEspinete1.peloEspinete\" 0.0;")
-backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Pelo Espinete" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "peloEspinete1.peloEspinete" );
       connectControl $slider ( "peloEspinete1.peloEspinete" );

       button -label "-Reset-" -command ("setAttr
\"cabezaMarsAttacks1.cabezaMarsAttacks\" 0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Cabeza Mars Attacks" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "cabezaMarsAttacks1.cabezaMarsAttacks" );
       connectControl $slider ( "cabezaMarsAttacks1.cabezaMarsAttacks" );


       button -label "-Reset-" -command ("setAttr \"peloCenicero1.peloCenicero\" 0.0;")
-backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Pelo Cenicero" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "peloCenicero1.peloCenicero" );
       connectControl $slider ( "peloCenicero1.peloCenicero" );

       button -label "-Reset-" -command ("setAttr \"peloCepillo1.peloCepillo\" 0.0;") -
backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Pelo Cepillo" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "peloCepillo1.peloCepillo" );
       connectControl $slider ( "peloCepillo1.peloCepillo" );

       button -label "-Reset-" -command ("setAttr \"peloTupe1.peloTupe\" 0.0;") -
backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Pelo Tupe" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "peloTupe1.peloTupe" );
       connectControl $slider ( "peloTupe1.peloTupe" );

       button -label "-Reset-" -command ("setAttr \"cabezaRoswell1.cabezaRoswell\"
0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Cabeza Roswell" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
```

```
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "cabezaRoswell1.cabezaRoswell" );
       connectControl $slider ( "cabezaRoswell1.cabezaRoswell" );

       button -label "-Reset-" -command ("setAttr \"peloTechnopunk1.peloTechnopunk\"
0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Pelo Technopunk" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "peloTechnopunk1.peloTechnopunk" );
       connectControl $slider ( "peloTechnopunk1.peloTechnopunk" );

       button -label "-Reset-" -command ("setAttr \"peloAfro1.peloAfro\" 0.0;") -
backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Pelo Afro" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "peloAfro1.peloAfro" );
       connectControl $slider ( "peloAfro1.peloAfro" );

       button -label "-Reset-" -command ("setAttr \"cabezaAliens1.cabezaAliens\" 0.0;")
-backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Cabeza Aliens" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "cabezaAliens1.cabezaAliens" );
       connectControl $slider ( "cabezaAliens1.cabezaAliens" );









   setParent ..;
   setParent ..;


   //--------------//
   //SLIDERS CARA  //
   //--------------//

   frameLayout -label "Modificaciones de Cara" FrameCambiosCara;
   rowColumnLayout -numberOfColumns 4 -cw 1 50 -cw 2 120 -cw 3 30 -cw 4 100;  //Layout
de los deslizadores para las shapes.

       button -label "-Reset-" -command ("setAttr \"ojosEstrechos1.ojosEstrechos\"
0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Ojos Estrechos" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "ojosEstrechos1.ojosEstrechos" );
       connectControl $slider ( "ojosEstrechos1.ojosEstrechos" );

       button -label "-Reset-" -command ("setAttr
\"ojosNarizEstrecho1.ojosNarizEstrecho\" 0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Ojos Nariz Estrechos" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "ojosNarizEstrecho1.ojosNarizEstrecho" );
       connectControl $slider ( "ojosNarizEstrecho1.ojosNarizEstrecho" );

       button -label "-Reset-" -command ("setAttr \"barbillaConica1.barbillaConica\"
0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "barbilla Conica" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "barbillaConica1.barbillaConica" );
       connectControl $slider ( "barbillaConica1.barbillaConica" );
```

```
        button -label "-Reset-" -command ("setAttr \"barbillaCuadrada1.barbillaCuadrada\"
0.0;") -backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Barbilla Cuadrada" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "barbillaCuadrada1.barbillaCuadrada" );
        connectControl $slider ( "barbillaCuadrada1.barbillaCuadrada" );

        button -label "-Reset-" -command ("setAttr \"barbillaSalida1.barbillaSalida\"
0.0;") -backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Barbilla Salida" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "barbillaSalida1.barbillaSalida" );
        connectControl $slider ( "barbillaSalida1.barbillaSalida" );

        button -label "-Reset-" -command ("setAttr \"bocaCorleone1.bocaCorleone\" 0.0;")
-backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Boca Corleone" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "bocaCorleone1.bocaCorleone" );
        connectControl $slider ( "bocaCorleone1.bocaCorleone" );

        button -label "-Reset-" -command ("setAttr \"bocaDescolgada1.bocaDescolgada\"
0.0;") -backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Boca Descolgada" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "bocaDescolgada1.bocaDescolgada" );
        connectControl $slider ( "bocaDescolgada1.bocaDescolgada" );

        button -label "-Reset-" -command ("setAttr \"bocaGrande1.bocaGrande\" 0.0;") -
backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Boca Grande" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "bocaGrande1.bocaGrande" );
        connectControl $slider ( "bocaGrande1.bocaGrande" );

        button -label "-Reset-" -command ("setAttr \"cejasRectas1.cejasRectas\" 0.0;") -
backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Cejas Rectas" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "cejasRectas1.cejasRectas" );
        connectControl $slider ( "cejasRectas1.cejasRectas" );

        button -label "-Reset-" -command ("setAttr \"cejasTristes1.cejasTristes\" 0.0;")
-backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Ceja Tristes" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "cejasTristes1.cejasTristes" );
        connectControl $slider ( "cejasTristes1.cejasTristes" );

        button -label "-Reset-" -command ("setAttr \"chupandoLimon1.chupandoLimon\"
0.0;") -backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Boca Chupando Limon" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "chupandoLimon1.chupandoLimon" );
        connectControl $slider ( "chupandoLimon1.chupandoLimon" );

        button -label "-Reset-" -command ("setAttr
\"mandibulaProminente1.mandibulaProminente\" 0.0;") -backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Mandibula Prominente" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
```

```
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "mandibulaProminente1.mandibulaProminente" );
       connectControl $slider ( "mandibulaProminente1.mandibulaProminente" );

       button -label "-Reset-" -command ("setAttr \"narizAguilenia1.narizAguilenia\"
0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Nariz Aguilenia" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "narizAguilenia1.narizAguilenia" );
       connectControl $slider ( "narizAguilenia1.narizAguilenia" );

       button -label "-Reset-" -command ("setAttr \"narizChata1.narizChata\" 0.0;") -
backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Nariz Chata" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "narizChata1.narizChata" );
       connectControl $slider ( "narizChata1.narizChata" );

       button -label "-Reset-" -command ("setAttr \"narizGrande1.narizGrande\" 0.0;") -
backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Nariz Grande" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "narizGrande1.narizGrande" );
       connectControl $slider ( "narizGrande1.narizGrande" );

       button -label "-Reset-" -command ("setAttr \"pomulosAltos1.pomulosAltos\" 0.0;")
-backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Pomulos Altos" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "pomulosAltos1.pomulosAltos" );
       connectControl $slider ( "pomulosAltos1.pomulosAltos" );

       button -label "-Reset-" -command ("setAttr \"pomulosGrandes1.pomulosGrandes\"
0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Pomulos Grandes" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "pomulosGrandes1.pomulosGrandes" );
       connectControl $slider ( "pomulosGrandes1.pomulosGrandes" );

       button -label "-Reset-" -command ("setAttr \"ojeras1.ojeras\" 0.0;") -
backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Ojeras" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "ojeras1.ojeras" );
       connectControl $slider ( "ojeras1.ojeras" );

       button -label "-Reset-" -command ("setAttr \"pocaPequenia1.pocaPequenia\" 0.0;")
-backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Boca Pequeña" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "pocaPequenia1.pocaPequenia" );
       connectControl $slider ( "pocaPequenia1.pocaPequenia" );




   setParent ..;
   setParent ..;

   //--------------//
   //SLIDERS TORSO//
   //--------------//
```

```
    frameLayout -label "Modificaciones de Torso" FrameCambiosTorso;
    rowColumnLayout -numberOfColumns 4 -cw 1 50 -cw 2 120 -cw 3 30 -cw 4 100;  //Layout
de los deslizadores para las shapes.


       button -label "-Reset-" -command ("setAttr \"caderasEstrechas1.caderasEstrechas\"
0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Caderas Estrechas" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "caderasEstrechas1.caderasEstrechas" );
       connectControl $slider ( "caderasEstrechas1.caderasEstrechas" );

       button -label "-Reset-" -command ("setAttr \"colaCoxis1.colaCoxis\" 0.0;") -
backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Cola Coxis" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "colaCoxis1.colaCoxis" );
       connectControl $slider ( "colaCoxis1.colaCoxis" );

       button -label "-Reset-" -command ("setAttr \"culoPequenio1.culoPequenio\" 0.0;")
-backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Culo Pequeño" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "culoPequenio1.culoPequenio" );
       connectControl $slider ( "culoPequenio1.culoPequenio" );

       button -label "-Reset-" -command ("setAttr \"lateralesGrandes1.lateralesGrandes\"
0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Laterales Grandes" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "lateralesGrandes1.lateralesGrandes" );
       connectControl $slider ( "lateralesGrandes1.lateralesGrandes" );

       button -label "-Reset-" -command ("setAttr
\"espaldasPequenias1.espaldasPequenias\" 0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Espaldas Pequenias" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "espaldasPequenias1.espaldasPequenias" );
       connectControl $slider ( "espaldasPequenias1.espaldasPequenias" );

       button -label "-Reset-" -command ("setAttr \"paqueteGrande1.paqueteGrande\"
0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Paquete Grande" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "paqueteGrande1.paqueteGrande" );
       connectControl $slider ( "paqueteGrande1.paqueteGrande" );

       button -label "-Reset-" -command ("setAttr \"culoGrande1.culoGrande\" 0.0;") -
backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Culo Grande" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "culoGrande1.culoGrande" );
       connectControl $slider ( "culoGrande1.culoGrande" );

       button -label "-Reset-" -command ("setAttr \"caderasAnchas1.caderasAnchas\"
0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Caderas Anchas" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "caderasAnchas1.caderasAnchas" );
       connectControl $slider ( "caderasAnchas1.caderasAnchas" );
```

```
       button -label "-Reset-" -command ("setAttr \"espaldasAnchas1.espaldasAnchas\"
0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Espaldas Anchas" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "espaldasAnchas1.espaldasAnchas" );
       connectControl $slider ( "espaldasAnchas1.espaldasAnchas" );

       button -label "-Reset-" -command ("setAttr \"pechoPequenio1.pechoPequenio\"
0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Pecho Pequenio" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "pechoPequenio1.pechoPequenio" );
       connectControl $slider ( "pechoPequenio1.pechoPequenio" );

       button -label "-Reset-" -command ("setAttr \"torsoGrande1.torsoGrande\" 0.0;") -
backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Torso Grande" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "torsoGrande1.torsoGrande" );
       connectControl $slider ( "torsoGrande1.torsoGrande" );

       button -label "-Reset-" -command ("setAttr \"barriga1.barriga\" 0.0;") -
backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Barriga" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "barriga1.barriga" );
       connectControl $slider ( "barriga1.barriga" );

       button -label "-Reset-" -command ("setAttr \"chepa1.chepa\" 0.0;") -
backgroundColor 1 1 1 -w 70;
       string $label = `text -label "chepa" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "chepa1.chepa" );
       connectControl $slider ( "chepa1.chepa" );

       button -label "-Reset-" -command ("setAttr
\"camisaArrugasLigeras1.camisaArrugasLigeras\" 0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Camisa Arrugas Ligeras" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "camisaArrugasLigeras1.camisaArrugasLigeras" );
       connectControl $slider ( "camisaArrugasLigeras1.camisaArrugasLigeras" );

       button -label "-Reset-" -command ("setAttr \"camisaLargaLisa1.camisaLargaLisa\"
0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Camisa Larga Lisa" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "camisaLargaLisa1.camisaLargaLisa" );
       connectControl $slider ( "camisaLargaLisa1.camisaLargaLisa" );



   setParent ..;
   setParent ..;

   //---------------//
   //SLIDERS BRAZOS //
   //---------------//

   frameLayout -label "Modificaciones de Brazos" FrameCambiosBrazos;
   rowColumnLayout -numberOfColumns 4 -cw 1 50 -cw 2 120 -cw 3 30 -cw 4 100;  //Layout
de los deslizadores para las shapes.
```

```
        button -label "-Reset-" -command ("setAttr \"antebrazoGrande1.antebrazoGrande\"
0.0;") -backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Antebrazo Grande" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "antebrazoGrande1.antebrazoGrande" );
        connectControl $slider ( "antebrazoGrande1.antebrazoGrande" );

        button -label "-Reset-" -command ("setAttr
\"antebrazoPequenio1.antebrazoPequenio\" 0.0;") -backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Antebrazo Pequenio" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "antebrazoPequenio1.antebrazoPequenio" );
        connectControl $slider ( "antebrazoPequenio1.antebrazoPequenio" );

        button -label "-Reset-" -command ("setAttr \"bicepsPequenios1.bicepsPequenios\"
0.0;") -backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Biceps Pequenios" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "bicepsPequenios1.bicepsPequenios" );
        connectControl $slider ( "bicepsPequenios1.bicepsPequenios" );

        button -label "-Reset-" -command ("setAttr \"bicepsGrandes1.bicepsGrandes\"
0.0;") -backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Biceps Grandes" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "bicepsGrandes1.bicepsGrandes" );
        connectControl $slider ( "bicepsGrandes1.bicepsGrandes" );

        button -label "-Reset-" -command ("setAttr \"brazosGrandes1.brazosGrandes\"
0.0;") -backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Brazos Grandes" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "brazosGrandes1.brazosGrandes" );
        connectControl $slider ( "brazosGrandes1.brazosGrandes" );

        button -label "-Reset-" -command ("setAttr \"brazosLargos1.brazosLargos\" 0.0;")
-backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Brazos Largos" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "brazosLargos1.brazosLargos" );
        connectControl $slider ( "brazosLargos1.brazosLargos" );

        button -label "-Reset-" -command ("setAttr \"brazosPequenios1.brazosPequenios\"
0.0;") -backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Brazos Pequenios" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "brazosPequenios1.brazosPequenios" );
        connectControl $slider ( "brazosPequenios1.brazosPequenios" );

        button -label "-Reset-" -command ("setAttr \"hombrosPequenios1.hombrosPequenios\"
0.0;") -backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Hombros Pequenios" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "hombrosPequenios1.hombrosPequenios" );
        connectControl $slider ( "hombrosPequenios1.hombrosPequenios" );

        button -label "-Reset-" -command ("setAttr \"hombrosGrandes1.hombrosGrandes\"
0.0;") -backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Hombros Grandes" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
```

```
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "hombrosGrandes1.hombrosGrandes" );
        connectControl $slider ( "hombrosGrandes1.hombrosGrandes" );

        button -label "-Reset-" -command ("setAttr
\"homenajeMikeMignola1.homenajeMikeMignola\" 0.0;") -backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Homenaje Mike Mignola" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "homenajeMikeMignola1.homenajeMikeMignola" );
        connectControl $slider ( "homenajeMikeMignola1.homenajeMikeMignola" );

        button -label "-Reset-" -command ("setAttr \"manosEspadas1.manosEspadas\" 0.0;")
-backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Manos Espadas" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "manosEspadas1.manosEspadas" );
        connectControl $slider ( "manosEspadas1.manosEspadas" );

        button -label "-Reset-" -command ("setAttr \"manosGarras1.manosGarras\" 0.0;") -
backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Manos Garras" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "manosGarras1.manosGarras" );
        connectControl $slider ( "manosGarras1.manosGarras" );

        button -label "-Reset-" -command ("setAttr \"manosGrandes1.manosGrandes\" 0.0;")
-backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Manos Grandes" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "manosGrandes1.manosGrandes" );
        connectControl $slider ( "manosGrandes1.manosGrandes" );

        button -label "-Reset-" -command ("setAttr \"manosSartenes1.manosSartenes\"
0.0;") -backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Manos Sartenes" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "manosSartenes1.manosSartenes" );
        connectControl $slider ( "manosSartenes1.manosSartenes" );

        button -label "-Reset-" -command ("setAttr \"puniosCerrados1.puniosCerrados\"
0.0;") -backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Punios Cerrados" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "puniosCerrados1.puniosCerrados" );
        connectControl $slider ( "puniosCerrados1.puniosCerrados" );

        button -label "-Reset-" -command ("setAttr \"tricepsGrandes1.tricepsGrandes\"
0.0;") -backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Triceps Grandes" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "tricepsGrandes1.tricepsGrandes" );
        connectControl $slider ( "tricepsGrandes1.tricepsGrandes" );

        button -label "-Reset-" -command ("setAttr \"tricepsPequenios1.tricepsPequenios\"
0.0;") -backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Triceps Pequenios" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "tricepsPequenios1.tricepsPequenios" );
        connectControl $slider ( "tricepsPequenios1.tricepsPequenios" );
```

```
       button -label "-Reset-" -command ("setAttr
\"cuchillasWolverine1.cuchillasWolverine\" 0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Cuchillas Wolverine" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "cuchillasWolverine1.cuchillasWolverine" );
       connectControl $slider ( "cuchillasWolverine1.cuchillasWolverine" );


       button -label "-Reset-" -command ("setAttr \"pinchosHombros1.pinchosHombros\"
0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Pinchos Hombros" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "pinchosHombros1.pinchosHombros" );
       connectControl $slider ( "pinchosHombros1.pinchosHombros" );


   setParent ..;
   setParent ..;

   //---------------//
   //SLIDERS PIERNAS//
   //---------------//

   frameLayout -label "Modificaciones de Piernas" FrameCambiosPiernas;
   rowColumnLayout -numberOfColumns 4 -cw 1 50 -cw 2 120 -cw 3 30 -cw 4 100;  //Layout
de los deslizadores para las shapes.

       button -label "-Reset-" -command ("setAttr \"botasElficas1.botasElficas\" 0.0;")
-backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Botas Elficas" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "botasElficas1.botasElficas" );
       connectControl $slider ( "botasElficas1.botasElficas" );

       button -label "-Reset-" -command ("setAttr \"botasMilitares1.botasMilitares\"
0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Botas Militares" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "botasMilitares1.botasMilitares" );
       connectControl $slider ( "botasMilitares1.botasMilitares" );

       button -label "-Reset-" -command ("setAttr
\"pantalonesHolgados1.pantalonesHolgados\" 0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Pantalones Holgados" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "pantalonesHolgados1.pantalonesHolgados" );
       connectControl $slider ( "pantalonesHolgados1.pantalonesHolgados" );

       button -label "-Reset-" -command ("setAttr \"pantalonesLisos1.pantalonesLisos\"
0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Pantalones Lisos" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "pantalonesLisos1.pantalonesLisos" );
       connectControl $slider ( "pantalonesLisos1.pantalonesLisos" );

       button -label "-Reset-" -command ("setAttr \"piernasPequenias1.piernasPequenias\"
0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Piernas Pequenias" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "piernasPequenias1.piernasPequenias" );
       connectControl $slider ( "piernasPequenias1.piernasPequenias" );
```

```
        button -label "-Reset-" -command ("setAttr \"gemelosAnchos1.gemelosAnchos\"
0.0;") -backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Gemelos Anchos" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "gemelosAnchos1.gemelosAnchos" );
        connectControl $slider ( "gemelosAnchos1.gemelosAnchos" );

        button -label "-Reset-" -command ("setAttr \"gemelosEstrechos1.gemelosEstrechos\"
0.0;") -backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Gemelos Estrechos" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "gemelosEstrechos1.gemelosEstrechos" );
        connectControl $slider ( "gemelosEstrechos1.gemelosEstrechos" );

        button -label "-Reset-" -command ("setAttr \"muslosAnchos1.muslosAnchos\" 0.0;")
-backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Muslos Anchos" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "muslosAnchos1.muslosAnchos" );
        connectControl $slider ( "muslosAnchos1.muslosAnchos" );

        button -label "-Reset-" -command ("setAttr \"piesGarras1.piesGarras\" 0.0;") -
backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Pies Garras" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "piesGarras1.piesGarras" );
        connectControl $slider ( "piesGarras1.piesGarras" );

        button -label "-Reset-" -command ("setAttr \"piesGrandes1.piesGrandes\" 0.0;") -
backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Pies Grandes" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "piesGrandes1.piesGrandes" );
        connectControl $slider ( "piesGrandes1.piesGrandes" );

        button -label "-Reset-" -command ("setAttr \"pinchosMuslos1.pinchosMuslos\"
0.0;") -backgroundColor 1 1 1 -w 70;
        string $label = `text -label "Pinchos Muslos" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "pinchosMuslos1.pinchosMuslos" );
        connectControl $slider ( "pinchosMuslos1.pinchosMuslos" );

        button -label "-Reset-" -command ("setAttr \"pinchoRodilla1.pinchoRodilla\"
0.0;") -backgroundColor 1 1 1 -w 70;
        string $label = `text -label "pinchoRodilla" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "pinchoRodilla1.pinchoRodilla" );
        connectControl $slider ( "pinchoRodilla1.pinchoRodilla" );

        button -label "-Reset-" -command ("setAttr \"pinchoTobillo1.pinchoTobillo\"
0.0;") -backgroundColor 1 1 1 -w 70;
        string $label = `text -label "PinchoTobillo" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
        // Conecta una barra de desplazamiento con la shape del modelo
        connectControl $field ( "pinchoTobillo1.pinchoTobillo" );
        connectControl $slider ( "pinchoTobillo1.pinchoTobillo" );

        button -label "-Reset-" -command ("setAttr \"muslosEstrechos1.muslosEstrechos\"
0.0;") -backgroundColor 1 1 1 -w 70;
        string $label = `text -label "muslosEstrechos" -align "center"`;
        string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
        string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
```

```
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "muslosEstrechos1.muslosEstrechos" );
       connectControl $slider ( "muslosEstrechos1.muslosEstrechos" );

       button -label "-Reset-" -command ("setAttr \"babuchas1.babuchas\" 0.0;") -
backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Babuchas" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "babuchas1.babuchas" );
       connectControl $slider ( "babuchas1.babuchas" );

       button -label "-Reset-" -command ("setAttr \"espolonTobillo1.espolonTobillo\"
0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Espolon Tobillo" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "espolonTobillo1.espolonTobillo" );
       connectControl $slider ( "espolonTobillo1.espolonTobillo" );

       button -label "-Reset-" -command ("setAttr \"pezunia1.pezunia\" 0.0;") -
backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Pezunia" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "pezunia1.pezunia" );
       connectControl $slider ( "pezunia1.pezunia" );


   setParent ..;
   setParent ..;

   //------------------//
   //SLIDERS ESPECIALES//
   //------------------//

   frameLayout -label "Modificaciones Especiales" FrameCambiosEspeciales;
   rowColumnLayout -numberOfColumns 4 -cw 1 50 -cw 2 120 -cw 3 30 -cw 4 100;  //Layout
de los deslizadores para las shapes.

       button -label "-Reset-" -command ("setAttr \"alto1.alto\" 0.0;") -backgroundColor
1 1 1 -w 70;
       string $label = `text -label "Alto" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "alto1.alto" );
       connectControl $slider ( "alto1.alto" );

       button -label "-Reset-" -command ("setAttr \"enano1.enano\" 0.0;") -
backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Enano" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "enano1.enano" );
       connectControl $slider ( "enano1.enano" );

       button -label "-Reset-" -command ("setAttr \"esqueleto1.esqueleto\" 0.0;") -
backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Esqueleto" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "esqueleto1.esqueleto" );
       connectControl $slider ( "esqueleto1.esqueleto" );

       button -label "-Reset-" -command ("setAttr
\"normalizaSinParallel1.normalizaSinParallel\" 0.0;") -backgroundColor 1 1 1 -w 70;
       string $label = `text -label "Suavizar todas las shapes" -align "center"`;
       string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
       string $slider = `floatSlider -min 0.0 -max 1.0 -width 1`;
       // Conecta una barra de desplazamiento con la shape del modelo
       connectControl $field ( "normalizaSinParallel1.normalizaSinParallel" );
```

```
       connectControl $slider ( "normalizaSinParallel1.normalizaSinParallel" );




    setParent ..;
    setParent ..;

    string $Directorio;
    $fileId=`fopen "C:/hlocal/muniecoconfig.txt" "r"`;
    $Directorio = `fgetline $fileId`;
    fclose $fileId;

    columnLayout -w 1000;
    string $exportashapes = "select -r Dummy;" + "file -op \"v=0\" -typ \"mayaAscii\" -es
\""+$Directorio+"backupmalla.ma\";" ;
    $exportashapes=$exportashapes+"select -cl;";
    button -label "Generar Back-Up Malla" -command ($exportashapes) -w 300;
    button -label "Cargar Back-Up Anterior" -command ("importaMalla") -w 300;
    //string $slider = `floatSlider2 -min 0.0 -max 1.0 -width 1`;
      //global float $nivel;
    //float $a;
    //$nivel=0.1;
    //connectControl $slider ($nivel);


    frameLayout -label "Generación Aleatoria" FrameCambiosEspeciales;
    global int $s = 1000;
    $s = rand (1000);


       columnLayout;  //Layout de los deslizadores para las shapes.
          button -label " [ Personaje aleatorio ] " -command ("personajeAleatorio") -w
300;
          rowColumnLayout -numberOfColumns 3;  //Layout de los deslizadores para las
shapes.

             text -label "     + Deformacion";
             floatSlider -min 0.0 -max 1.0 -width 1 grado;
             global float $nivel = 0.0;
             text -label "     - Deformacion";

          setParent ..;
        setParent ..;

    setParent ..;






setParent ..;









       //Obtengo el ID del workspace actual:
       string $mayaWorkspace = `workspace -q -dir`;
       // Establezco como workspace actual el directorio configurado por el usuario:
       workspace -dir $Directorio ;
       workspace -o $Directorio;

       $Directorio1 = $Directorio + "scenes/EscenaProyecto.mb";
       string $Directorio2 = $Directorio + "scenes/EscenaNueva.mb";



       /*string $respuesta=`confirmDialog -message "Aviso: El plug-in tardará un tiempo
en cargarse\nPor favor, sea paciente" -button "OK" -button "Cancelar"
             -defaultButton "OK" -cancelButton "Cancelar" -dismissString "Cancelar"`;
```

```
       if ($respuesta=="OK")
       {*/
              print("Ok makey");
           showWindow;

           print ("abro la buena\n");
           pv_performAction ($Directorio1) "Best Guess";
           file -f -options "v=0"  -typ "mayaBinary" -o ($Directorio1);
           addRecentFile($Directorio1, "mayaBinary");

           print ("salvo a la otra\n");
           /*pv_performAction $Directorio2 "mayaBinary";
           file -f -save -options "v=0" -type "mayaBinary";*/
           file -rename $Directorio2; file -save -type "mayaBinary";



           print ("abro la otra\n");
           file -f -options "v=0"  -typ "mayaBinary" -o
$Directorio2;addRecentFile($Directorio2, "mayaBinary");

           DisplayShadedAndTextured;

       /*

           }

       else {print("No se hace nada");} //Otra Rama del If general. De momento no
hacemos nada*/

}//Fin del Procedimiento para la ventana 1




//*****************************************************************************

global proc Ventana2()
{
   if ( `window -exists control2` ) deleteUI -window control2;

   string $window = `window -title "Ventana de Control" -rtf 1 control2`;
   string $scrollLayout = `scrollLayout -horizontalScrollBarThickness 16 -
verticalScrollBarThickness   16`; //Crea un scroll
   string $opcion; //Variable que usaremos para ver qué animación está seleccionada en
el combobox.
   string $opcion2; //Variable que usaremos para ver qué textura está seleccionada en el
combobox.

   global int $primeravez = 0; //Variable que nos indica si es la primera vez que vamos
a cargar una textura o no;

   select -r -ne character1;
   doImportClipArgList 4 { "1","1" } ;
   editMenuUpdate MayaWindow|mainEditMenu;
   ImportSkinWeightMaps;
   sets -e -forceElement MatDummy Dummy;



   columnLayout -cal "center" -w 300; //columna de botones

       string $Directorio, $Directorio1, $Directorio2, $Directorio3;
       $fileId=`fopen "C:/hlocal/muniecoconfig.txt" "r"`;
       $Directorio = `fgetline $fileId`;
       fclose $fileId;
       $Directorio1 = "source \"" + $Directorio + "Scripts/scriptJoints.mel\";";
       $Directorio2 = "source \"" + $Directorio + "Scripts/scriptAjustaHuesos2.mel\";";
       $Directorio3 = "source \"" + $Directorio +
"Scripts/scriptPesadoAutomatico.mel\";";
       $Directorio4 = "source \"" + $Directorio + "Scripts/scriptBorraShapes.mel\";";
       $Directorio5 = "source \"" + $Directorio + "Scripts/scriptBorraCapas.mel\";";

       string $desactiva1 = "button -e -en 0 b1";
       string $desactiva2 = "button -e -en 0 b2";
       string $desactiva3 = "button -e -en 0 b3";
```

```
       string $eliminashapes = "if ( `window -exists customizado` ) deleteUI -window
customizado;"+$desactiva1;


       //Boton que llama al script que crea los Locators
       button -label "Crea Referencias" -command ($Directorio1+"select -r
Dummy;"+"DeleteAllHistory;"+$Directorio4+$eliminashapes) -w 150 b1;

       string $desactiva1 = "button -e -en 0 b1";
       string $desactiva2 = "button -e -en 0 b2;";
       string $desactiva3 = "button -e -en 0 b3";
       string $importapesado = "select -r Dummy;";
       $importapesado = $importapesado + "importSkinWeightMap
\""+$Directorio+"sourceimages/pesado.weightMap\" \"map\";select -cl;";


       //Boton que llama al script que realiza el pesado del esqueleto con la malla
       button -label "Pesado Automático" -command
($Directorio2+$Directorio3+$desactiva2+$importapesado) -w 150 b2;

       frameLayout -label "Modificaciones de Color" -width 150 FrameCambiosColor;
          rowColumnLayout -numberOfColumns 3 -cw 1 20 -cw 2 30 -cw 3 100;  //Layout de
los deslizadores para las shapes.


              string $label = `text -label "R" -align "center"`;
              string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
              string $slider = `floatSlider -min 0.0 -max 1.0`;
              // Conecta una barra de desplazamiento con la shape del modelo
              connectControl $field ( "ramp1.colorEntryList[1].color.ecr");
              connectControl $slider ( "ramp1.colorEntryList[1].color.ecr" );

              string $label = `text -label "G" -align "center"`;
              string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
              string $slider = `floatSlider -min 0.0 -max 1.0`;
              // Conecta una barra de desplazamiento con la shape del modelo
              connectControl $field ( "ramp1.colorEntryList[1].color.ecg");
              connectControl $slider ( "ramp1.colorEntryList[1].color.ecg" );

              string $label = `text -label "B" -align "center"`;
              string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
              string $slider = `floatSlider -min 0.0 -max 1.0`;
              // Conecta una barra de desplazamiento con la shape del modelo
              connectControl $field ( "ramp1.colorEntryList[1].color.ecb");
              connectControl $slider ( "ramp1.colorEntryList[1].color.ecb" );




          setParent ..;
       setParent ..;

       frameLayout -label "Influencia Textura/Color" -width 150 FrameColorTextura;
          rowColumnLayout -numberOfColumns 3 -cw 1 20 -cw 2 30 -cw 3 100;  //Layout de
los deslizadores para las shapes.


              string $label = `text -label "R" -align "center"`;
              string $field = `floatField -min 0.0 -max 1.0 -precision 0.1`;
              string $slider = `floatSlider -min 0.0 -max 1.0`;
              // Conecta una barra de desplazamiento con la shape del modelo
              connectControl $field ( "blendColors1.blender");
              connectControl $slider ( "blendColors1.blender" );




          setParent ..;
       setParent ..;


       frameLayout -label "Modificaciones de Capas" -width 150 FrameCambiosCapas4;
          rowColumnLayout -numberOfColumns 1 -width 300 ;  //Layout de los checkboxs de
las capas de textura.
```

```
              string $comandoOn = "setAttr \"layeredTexture1.inputs[5].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[5].isVisible\" 0;";
              checkBox -label "Casco" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)   -
align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[6].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[6].isVisible\" 0;";
              checkBox -label "Traje Esp. Am." -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[7].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[7].isVisible\" 0;";
              checkBox -label "Traje Esp. Rojo" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[8].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[8].isVisible\" 0;";
              checkBox -label "Traje Esp. Negro" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[9].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[9].isVisible\" 0;";
              checkBox -label "Traje Esp. Blanco" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[10].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[10].isVisible\"
0;";
              checkBox -label "Protecciones" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[11].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[11].isVisible\"
0;";
              checkBox -label "Antibalas 2" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[12].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[12].isVisible\"
0;";
              checkBox -label "Antibalas 1" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[13].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[13].isVisible\"
0;";
              checkBox -label "Swat" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)   -align
"left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[14].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[14].isVisible\"
0;";
              checkBox -label "Pasamontañas" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[15].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[15].isVisible\"
0;";
              checkBox -label "Botas" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)   -
align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[16].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[16].isVisible\"
0;";
              checkBox -label "Spidey Custom" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[17].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[17].isVisible\"
0;";
              checkBox -label "Venom Custom" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[18].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[18].isVisible\"
0;";
```

```
               checkBox -label "Camuf. Polar" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[19].isVisible\" 1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[19].isVisible\"
0;";
               checkBox -label "Camuf. Sabana" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[20].isVisible\" 1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[20].isVisible\"
0;";
               checkBox -label "Camuf. Dia Lluvia" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[21].isVisible\" 1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[21].isVisible\"
0;";
               checkBox -label "Camuf. Jungla" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[22].isVisible\" 1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[22].isVisible\"
0;";
               checkBox -label "Malla Ninja" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[23].isVisible\" 1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[23].isVisible\"
0;";
               checkBox -label "Malla Az. Claro" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[24].isVisible\" 1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[24].isVisible\"
0;";
               checkBox -label "Malla Negra" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[25].isVisible\" 1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[25].isVisible\"
0;";
               checkBox -label "Malla Blanca" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[26].isVisible\" 1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[26].isVisible\"
0;";
               checkBox -label "Corbata Az. Oscuro" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[27].isVisible\" 1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[27].isVisible\"
0;";
               checkBox -label "Corbata Az. Claro" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[28].isVisible\" 1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[28].isVisible\"
0;";
               checkBox -label "Corbata Roja" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[29].isVisible\" 1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[29].isVisible\"
0;";
               checkBox -label "Corbata Negra" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[30].isVisible\" 1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[30].isVisible\"
0;";
               checkBox -label "Corbata Blanca" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[31].isVisible\" 1;";
```

```
              string $comandoOff = "setAttr \"layeredTexture1.inputs[31].isVisible\"
0;";
              checkBox -label "Camisa MC N" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[32].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[32].isVisible\"
0;";
              checkBox -label "Camisa MC Gris" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[33].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[33].isVisible\"
0;";
              checkBox -label "Camisa MC Rs" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[34].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[34].isVisible\"
0;";
              checkBox -label "Camisa MC B" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[35].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[35].isVisible\"
0;";
              checkBox -label "Camisa ML Gr" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[36].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[36].isVisible\"
0;";
              checkBox -label "Camisa ML Rs" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[37].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[37].isVisible\"
0;";
              checkBox -label "Camisa ML N" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[38].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[38].isVisible\"
0;";
              checkBox -label "Camisa ML B" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[39].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[39].isVisible\"
0;";
              checkBox -label "Cam. Tir. Roja" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[40].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[40].isVisible\"
0;";
              checkBox -label "Cam. Tir. Marrón" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[41].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[41].isVisible\"
0;";
              checkBox -label "Cam. Tir. Verde" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[42].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[42].isVisible\"
0;";
              checkBox -label "Cam. Tir. Negra" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[43].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[43].isVisible\"
0;";
              checkBox -label "Cam. Tir. Blanca" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";
```

```
              string $comandoOn = "setAttr \"layeredTexture1.inputs[44].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[44].isVisible\"
0;";
              checkBox -label "Cam. M L Azul" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[45].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[45].isVisible\"
0;";
              checkBox -label "Cam. M L Verde" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[46].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[46].isVisible\"
0;";
              checkBox -label "Cam. M L Negra" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[47].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[47].isVisible\"
0;";
              checkBox -label "Cam. M L Blanca" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[48].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[48].isVisible\"
0;";
              checkBox -label "Camiseta Roja" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[49].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[49].isVisible\"
0;";
              checkBox -label "Camiseta Rosa" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[50].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[50].isVisible\"
0;";
              checkBox -label "Camiseta Pistacho" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[51].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[51].isVisible\"
0;";
              checkBox -label "Cam. Azul Oscuro" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[52].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[52].isVisible\"
0;";
              checkBox -label "Camiseta Negra" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[53].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[53].isVisible\"
0;";
              checkBox -label "Camiseta Blanca" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[54].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[54].isVisible\"
0;";
              checkBox -label "P. Cortos Ocre" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[55].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[55].isVisible\"
0;";
              checkBox -label "P. Cort. Pistacho" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[56].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[56].isVisible\"
0;";
              checkBox -label "P. Cortos Az." -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";
```

```
                string $comandoOn = "setAttr \"layeredTexture1.inputs[57].isVisible\" 1;";
                string $comandoOff = "setAttr \"layeredTexture1.inputs[57].isVisible\"
0;";
                checkBox -label "P. Cortos N." -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

                string $comandoOn = "setAttr \"layeredTexture1.inputs[58].isVisible\" 1;";
                string $comandoOff = "setAttr \"layeredTexture1.inputs[58].isVisible\"
0;";
                checkBox -label "P. Cortos B." -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

                string $comandoOn = "setAttr \"layeredTexture1.inputs[59].isVisible\" 1;";
                string $comandoOff = "setAttr \"layeredTexture1.inputs[59].isVisible\"
0;";
                checkBox -label "Pant. Azul Oscuro" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

                string $comandoOn = "setAttr \"layeredTexture1.inputs[60].isVisible\" 1;";
                string $comandoOff = "setAttr \"layeredTexture1.inputs[60].isVisible\"
0;";
                checkBox -label "Pant. Azul Claro" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

                string $comandoOn = "setAttr \"layeredTexture1.inputs[61].isVisible\" 1;";
                string $comandoOff = "setAttr \"layeredTexture1.inputs[61].isVisible\"
0;";
                checkBox -label "Pant. Pistacho" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

                string $comandoOn = "setAttr \"layeredTexture1.inputs[62].isVisible\" 1;";
                string $comandoOff = "setAttr \"layeredTexture1.inputs[62].isVisible\"
0;";
                checkBox -label "Pant. Negros" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

                string $comandoOn = "setAttr \"layeredTexture1.inputs[63].isVisible\" 1;";
                string $comandoOff = "setAttr \"layeredTexture1.inputs[63].isVisible\"
0;";
                checkBox -label "Pant. Blancos" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

                string $comandoOn = "setAttr \"layeredTexture1.inputs[64].isVisible\" 1;";
                string $comandoOff = "setAttr \"layeredTexture1.inputs[64].isVisible\"
0;";
                checkBox -label "Calce. Cortos N." -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

                string $comandoOn = "setAttr \"layeredTexture1.inputs[65].isVisible\" 1;";
                string $comandoOff = "setAttr \"layeredTexture1.inputs[65].isVisible\"
0;";
                checkBox -label "Calce. Cortos B." -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

                /*string $comandoOn = "setAttr \"layeredTexture1.inputs[66].isVisible\"
1;";
                string $comandoOff = "setAttr \"layeredTexture1.inputs[66].isVisible\"
0;";
                checkBox -label "Calce. Larg. Blancos" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";*/

                string $comandoOn = "setAttr \"layeredTexture1.inputs[67].isVisible\" 1;";
                string $comandoOff = "setAttr \"layeredTexture1.inputs[67].isVisible\"
0;";
                checkBox -label "Calce. Larg. Blancos" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

                /*string $comandoOn = "setAttr \"layeredTexture1.inputs[68].isVisible\"
1;";
                string $comandoOff = "setAttr \"layeredTexture1.inputs[68].isVisible\"
0;";
                checkBox -label "Zapatos Negros" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";*/

                string $comandoOn = "setAttr \"layeredTexture1.inputs[69].isVisible\" 1;";
```

```
               string $comandoOff = "setAttr \"layeredTexture1.inputs[69].isVisible\"
0;";
               checkBox -label "Zapatos Negros" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[70].isVisible\" 1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[70].isVisible\"
0;";
               checkBox -label "Zapatos Blancos" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[71].isVisible\" 1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[71].isVisible\"
0;";
               checkBox -label "Gafas Sol Negras" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[72].isVisible\" 1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[72].isVisible\"
0;";
               checkBox -label "Gafas Sol Marrones" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[73].isVisible\" 1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[73].isVisible\"
0;";
               checkBox -label "Cuernos Blancos" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[74].isVisible\" 1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[74].isVisible\"
0;";
               checkBox -label "Cicatriz" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)   -
align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[75].isVisible\" 1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[75].isVisible\"
0;";
               checkBox -label "Barba 2 Días" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[76].isVisible\" 1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[76].isVisible\"
0;";
               checkBox -label "Barba Patillas N." -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[77].isVisible\" 1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[77].isVisible\"
0;";
               checkBox -label "Barba Lobezno N." -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";


               string $comandoOn = "setAttr \"layeredTexture1.inputs[78].isVisible\" 1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[78].isVisible\"
0;";
               checkBox -label "Barba Cerrada N." -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[79].isVisible\" 1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[79].isVisible\"
0;";
               checkBox -label "Perilla Chino" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[80].isVisible\" 1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[80].isVisible\"
0;";
               checkBox -label "Mosca Negra" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[81].isVisible\" 1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[81].isVisible\"
0;";
               checkBox -label "Big. Bronson Blanco" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";
```

```
              string $comandoOn = "setAttr \"layeredTexture1.inputs[82].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[82].isVisible\"
0;";
              checkBox -label "Barba Señorial" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[83].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[83].isVisible\"
0;";
              checkBox -label "Big. Charles Bronson" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[84].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[84].isVisible\"
0;";
              checkBox -label "Cejas" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)   -
align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[85].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[85].isVisible\"
0;";
              checkBox -label "Pelo B. Entero" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[86].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[86].isVisible\"
0;";
              checkBox -label "Pelo N. Calvo" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[87].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[87].isVisible\"
0;";
              checkBox -label "Pelo N. Completo" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[88].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[88].isVisible\"
0;";
              checkBox -label "Piel Alien" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[89].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[89].isVisible\"
0;";
              checkBox -label "Piel Roja 3" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[90].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[90].isVisible\"
0;";
              checkBox -label "Piel Roja 2" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[91].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[91].isVisible\"
0;";
              checkBox -label "Piel Roja 1" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[92].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[92].isVisible\"
0;";
              checkBox -label "Piel Verde 3" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[93].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[93].isVisible\"
0;";
              checkBox -label "Piel Verde 2" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

              string $comandoOn = "setAttr \"layeredTexture1.inputs[94].isVisible\" 1;";
              string $comandoOff = "setAttr \"layeredTexture1.inputs[94].isVisible\"
0;";
```

```
               checkBox -label "Piel Verde 1" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[95].isVisible\" 1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[95].isVisible\"
0;";
               checkBox -label "Piel Rosada 3" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[96].isVisible\" 1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[96].isVisible\"
0;";
               checkBox -label "Piel Rosada 2" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[97].isVisible\" 1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[97].isVisible\"
0;";
               checkBox -label "Piel Rosada 1" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[98].isVisible\" 1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[98].isVisible\"
0;";
               checkBox -label "Piel Caucásica 3" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[99].isVisible\" 1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[99].isVisible\"
0;";
               checkBox -label "Piel Caucásica 2" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[100].isVisible\"
1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[100].isVisible\"
0;";
               checkBox -label "Piel Caucásica" -v 0 -onc ($comandoOn)  -ofc
($comandoOff)   -align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[101].isVisible\"
1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[101].isVisible\"
0;";
               checkBox -label "Piel Negra 3" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[102].isVisible\"
1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[102].isVisible\"
0;";
               checkBox -label "Piel Negra 2" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

               /*string $comandoOn = "setAttr \"layeredTexture1.inputs[103].isVisible\"
1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[103].isVisible\"
0;";
               checkBox -label "Piel Negra 1" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[104].isVisible\"
1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[104].isVisible\"
0;";
               checkBox -label "Piel Morena 3" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";*/

               string $comandoOn = "setAttr \"layeredTexture1.inputs[105].isVisible\"
1;";
               string $comandoOff = "setAttr \"layeredTexture1.inputs[105].isVisible\"
0;";
               checkBox -label "Piel Negra 1" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";

               string $comandoOn = "setAttr \"layeredTexture1.inputs[106].isVisible\"
1;";
```

```
               string $comandoOff = "setAttr \"layeredTexture1.inputs[106].isVisible\"
0;";
               checkBox -label "Piel Morena" -v 0 -onc ($comandoOn)  -ofc ($comandoOff)
-align "left";


          setParent ..;
       setParent ..;



       button -label "Preview Cuerpo" -command ("rendear;") -w 150 bp1;

       button -label "Preview Cara" -command ("rendear2;") -w 150 bp2;


       string $cueceTextura = "convertSolidTx -antiAlias 1 -bm 1 -fts 1 -sp 0 -sh 0 -
alpha 0 -doubleSided 0 ";
       $cueceTextura = $cueceTextura + "-componentRange 0 -resolutionX 512 -resolutionY
512 -fileFormat \"tga\"";
       $cueceTextura = $cueceTextura + "  layeredTexture1.outColor Dummy;";
       $asignainitial = "select -r Dummy ;sets -e -forceElement initialShadingGroup
Dummy;";
       string $com = $cueceTextura+$asignainitial+$desactiva3;
       button -label "Crear Textura" -command ($com) -w 150 b3;


       //Layout para la elección y carga de las texturas
       frameLayout -width 150 -height 62 -label "Animaciones" FrameAnimaciones;
          columnLayout ;
              optionMenu -cc "select -r -ne character1;" -w 100 menuanimaciones;

                  menuItem -label "iddle";

                  menuItem -label "andar";

                  menuItem -label "Andar_Alegre";

                  menuItem -label "Andar_Atras";

                  menuItem -label "andar_atras_2";

                  menuItem -label "Andar_Cojeando";

                  menuItem -label "andar_cojo";

                  menuItem -label "Andar_Marcial_Fascista";

                  menuItem -label "Andar_Marcial_Ruso";

                  menuItem -label "Andar_Poseido";

                  menuItem -label "andar_derecha";

                  menuItem -label "andar_izquierda";

                  menuItem -label "andar_gorilesco";

                  menuItem -label "andar_sigiloso";

                  menuItem -label "correr";

                  menuItem -label "Correr_Digno";

                  menuItem -label "Correr_Footing";

                  menuItem -label "agacharse";

                  menuItem -label "puñetazo_gancho";

                  menuItem -label "saltar";

                  menuItem -label "apuntar_pistola";

                  menuItem -label "disparar_pistola";

                  menuItem -label "coger";
```

```
                menuItem -label "hablar";

                menuItem -label "muerte";




          //Boton que añade la animación seleccionada en el combobox anterior
          button -label "ImportarAnimacion" -command ("currentTime 1;importaclip;") -w
150;

          setParent ..;
             setParent ..;

   string $muestraclips = "toggleAutoLoad clipEditorPanel1ClipEditor 1;";
   button -label "Mostrar Clips" -command ("CharacterAnimationEditor;select -r -ne
character1;"+$muestraclips) -w 150;


   string $agrupa = "select -r Dummy;group;xform -os -piv 0 0 0;select -r group1;rename
\"group1\" \"model\";reorder -relative -1 model;";
   string $despreview = "button -e -en 0 bp1;"+"button -e -en 0 bp2";

   //Boton que abre el menú de guardar
   button -label "Preparar Para Exportar" -command ("disconnectAttr blendColors1.output
lambert8.color;"+$Directorio5+$agrupa+$despreview) -w 150;


   //Boton que abre el menú de guardar
   button -label "Salvar" -command ("SaveSceneAs") -w 150;

   //Boton que cierra la ventanita
   button -label "Close" -command ("deleteUI -window " + $window) -w 150;


   setParent ..;
   showWindow;

}//Fin del Procedimiento Ventana2



//Procedimiento que importa la animacion seleccionada en el combobox.
global proc importaclip()
{
   select -r -ne character1;
   $opcion = `optionMenu  -query -value menuanimaciones`;

   //string $path = "I:/Clips/"+$opcion+".ma";
   //print ($path + "\n");

   string $Directorio, $Directorio1;
   $fileId=`fopen "C:/hlocal/muniecoconfig.txt" "r"`;
   $Directorio = `fgetline $fileId`;
   fclose $fileId;
   $Directorio1 = $Directorio + "clips/" + $opcion + ".ma";


   currentTime 1 ;

   clipEditorImportClip $Directorio1 "ma";

   select -r -ne character1 ;

};

//Procedimiento que carga la textura seleccionada en el combobox.
global proc cargatextura()
{

   $opcion = `optionMenu  -query -value menutextura`;

   defaultNavigation -connectToExisting -force true -source $opcion -destination
"MatDummy";
   connectAttr -f file1.outColor lambert6.color;
```

```
    string $path = "I:/"+$opcion+".jpg";
    print ($path + "\n");









};

global proc rendear()
{
    RenderViewWindow;
    renderWindowRenderCamera render renderView camara1;
}

global proc rendear2()
{
    RenderViewWindow;
    renderWindowRenderCamera render renderView camara2;
}

global proc importaMalla()
{
    select -r Dummy;
    delete;
    string $Directorio;
    $fileId=`fopen "C:/hlocal/muniecoconfig.txt" "r"`;
    $Directorio = `fgetline $fileId`;
    fclose $fileId;
    file -import -type "mayaAscii" -rpr "backupmalla" -options "v=0"
($Directorio+"backupmalla.ma");
    sets -e -forceElement MatDummy Dummy;
    if ( `window -exists customizado` ) deleteUI -window customizado;




}


//Procedimiento encargado de establecer un random a cada uno de los modificadores de la
malla de personaje
global proc personajeAleatorio()
{
global int $s;
$s = $s + 7;
seed $s;
global float $nivel;
$nivel = `floatSlider -q -value grado`;
//$nivel=0.7;
setAttr "orejasElficas1.orejasElficas" 0;
setAttr "orejasGordas1.orejasGordas" 0;
setAttr "orejasGrandes1.orejasGrandes" 0;
setAttr "orejasNormales1.orejasNormales" 0;
setAttr "orejasPequenias1.orejasPequenias" 0;
setAttr "orejasYoda1.orejasYoda" 0;
setAttr "cuelloCorto1.cuelloCorto" 0;
setAttr "cuelloEstrecho1.cuelloEstrecho" 0;
setAttr "cuelloLargo1.cuelloLargo" 0;
setAttr "cuelloAncho1.cuelloAncho" 0;
setAttr "cuernosAtras1.cuernosAtras" 0;
setAttr "cuernosBajos1.cuernosBajos" 0;
setAttr "cuernosBolas1.cuernosBolas" 0;
setAttr "cuernosBultos1.cuernosBultos" 0;
setAttr "cuernosCerrados1.cuernosCerrados" 0;
setAttr "cuernosDelanteros1.cuernosDelanteros" 0;
setAttr "cuernosGrandes1.cuernosGrandes" 0;
setAttr "cuernosParabolicos1.cuernosParabolicos" 0;
setAttr "orejasCuernosArriba1.orejasCuernosArriba" 0;
setAttr "orejasRadar1.orejasRadar" 0;
setAttr "orejasTrompeta1.orejasTrompeta" 0;
setAttr "cabezaGrande1.cabezaGrande" 0;
setAttr "cabezaAlargada1.cabezaAlargada" 0;
```

```
setAttr "cabezaCuadrada1.cabezaCuadrada" 0;
setAttr "cabezaPequenia1.cabezaPequenia" 0;
setAttr "cascoMilitar1.cascoMilitar" 0;
setAttr "crestaPunk1.crestaPunk" 0;
setAttr "cuernosCortados1.cuernosCortados" 0;
setAttr "peloEspinete1.peloEspinete" 0;
setAttr "cabezaMarsAttacks1.cabezaMarsAttacks" 0;
setAttr "peloCenicero1.peloCenicero" 0;
setAttr "peloCepillo1.peloCepillo" 0;
setAttr "peloTupe1.peloTupe" 0;
setAttr "cabezaRoswell1.cabezaRoswell" 0;
setAttr "peloTechnopunk1.peloTechnopunk" 0;
setAttr "peloAfro1.peloAfro" 0;
setAttr "cabezaAliens1.cabezaAliens" 0;
setAttr "ojosEstrechos1.ojosEstrechos" 0;
setAttr "ojosNarizEstrecho1.ojosNarizEstrecho" 0;
setAttr "barbillaConica1.barbillaConica" 0;
setAttr "barbillaCuadrada1.barbillaCuadrada" 0;
setAttr "barbillaSalida1.barbillaSalida" 0;
setAttr "bocaCorleone1.bocaCorleone" 0;
setAttr "bocaDescolgada1.bocaDescolgada" 0;
setAttr "bocaGrande1.bocaGrande" 0;
setAttr "cejasRectas1.cejasRectas" 0;
setAttr "cejasTristes1.cejasTristes" 0;
setAttr "chupandoLimon1.chupandoLimon" 0;
setAttr "mandibulaProminente1.mandibulaProminente" 0;
setAttr "narizAguilenia1.narizAguilenia" 0;
setAttr "narizChata1.narizChata" 0;
setAttr "narizGrande1.narizGrande" 0;
setAttr "pomulosAltos1.pomulosAltos" 0;
setAttr "pomulosGrandes1.pomulosGrandes" 0;
setAttr "ojeras1.ojeras" 0;
setAttr "pocaPequenia1.pocaPequenia" 0;
setAttr "caderasEstrechas1.caderasEstrechas" 0;
setAttr "colaCoxis1.colaCoxis" 0;
setAttr "culoPequenio1.culoPequenio" 0;
setAttr "lateralesGrandes1.lateralesGrandes" 0;
setAttr "espaldasPequenias1.espaldasPequenias" 0;
setAttr "paqueteGrande1.paqueteGrande" 0;
setAttr "culoGrande1.culoGrande" 0;
setAttr "caderasAnchas1.caderasAnchas" 0;
setAttr "espaldasAnchas1.espaldasAnchas" 0;
setAttr "pechoPequenio1.pechoPequenio" 0;
setAttr "torsoGrande1.torsoGrande" 0;
setAttr "barriga1.barriga" 0;
setAttr "chepa1.chepa" 0;
setAttr "camisaArrugasLigeras1.camisaArrugasLigeras" 0;
setAttr "camisaLargaLisa1.camisaLargaLisa" 0;
setAttr "antebrazoGrande1.antebrazoGrande" 0;
setAttr "antebrazoPequenio1.antebrazoPequenio" 0;
setAttr "bicepsPequenios1.bicepsPequenios" 0;
setAttr "bicepsGrandes1.bicepsGrandes" 0;
setAttr "brazosGrandes1.brazosGrandes" 0;
setAttr "brazosLargos1.brazosLargos" 0;
setAttr "brazosPequenios1.brazosPequenios" 0;
setAttr "hombrosPequenios1.hombrosPequenios" 0;
setAttr "hombrosGrandes1.hombrosGrandes" 0;
setAttr "homenajeMikeMignola1.homenajeMikeMignola" 0;
setAttr "manosEspadas1.manosEspadas" 0;
setAttr "manosGarras1.manosGarras" 0;
setAttr "manosGrandes1.manosGrandes" 0;
setAttr "manosSartenes1.manosSartenes" 0;
setAttr "puniosCerrados1.puniosCerrados" 0;
setAttr "tricepsGrandes1.tricepsGrandes" 0;
setAttr "tricepsPequenios1.tricepsPequenios" 0;
setAttr "cuchillasWolverine1.cuchillasWolverine" 0;
setAttr "pinchosHombros1.pinchosHombros" 0;
setAttr "botasElficas1.botasElficas" 0;
setAttr "botasMilitares1.botasMilitares" 0;
setAttr "pantalonesHolgados1.pantalonesHolgados" 0;
setAttr "pantalonesLisos1.pantalonesLisos" 0;
setAttr "piernasPequenias1.piernasPequenias" 0;
setAttr "gemelosAnchos1.gemelosAnchos" 0;
setAttr "gemelosEstrechos1.gemelosEstrechos" 0.0;
setAttr "muslosAnchos1.muslosAnchos" 0;
setAttr "piesGarras1.piesGarras" 0;
setAttr "piesGrandes1.piesGrandes" 0;
```

```
setAttr "pinchosMuslos1.pinchosMuslos" 0;
setAttr "pinchoRodilla1.pinchoRodilla" 0;
setAttr "pinchoTobillo1.pinchoTobillo" 0;
setAttr "muslosEstrechos1.muslosEstrechos" 0;
setAttr "babuchas1.babuchas" 0;
setAttr "espolonTobillo1.espolonTobillo" 0;
setAttr "pezunia1.pezunia" 0;
setAttr "alto1.alto" 0;
setAttr "enano1.enano" 0;
setAttr "esqueleto1.esqueleto" 0;
setAttr "normalizaSinParallel1.normalizaSinParallel" 0;


if (rand(1) > $nivel) setAttr ("orejasElficas1.orejasElficas",rand(1));
if (rand(1) > $nivel) setAttr ("orejasGordas1.orejasGordas",rand(1));
if (rand(1) > $nivel) setAttr ("orejasGrandes1.orejasGrandes",rand(1));
if (rand(1) > $nivel) setAttr ("orejasNormales1.orejasNormales",rand(1));
if (rand(1) > $nivel) setAttr ("orejasPequenias1.orejasPequenias",rand(1));
if (rand(1) > $nivel) setAttr ("orejasYoda1.orejasYoda",rand(1));
if (rand(1) > $nivel) setAttr ("cuelloCorto1.cuelloCorto",rand(1));
if (rand(1) > $nivel) setAttr ("cuelloEstrecho1.cuelloEstrecho",rand(1));
if (rand(1) > $nivel) setAttr ("cuelloLargo1.cuelloLargo",rand(1));
if (rand(1) > $nivel) setAttr ("cuelloAncho1.cuelloAncho",rand(1));
if (rand(1) > $nivel) setAttr ("cuernosAtras1.cuernosAtras",rand(1));
if (rand(1) > $nivel) setAttr ("cuernosBajos1.cuernosBajos",rand(1));
if (rand(1) > $nivel) setAttr ("cuernosBolas1.cuernosBolas" ,rand(1));
if (rand(1) > $nivel) setAttr ("cuernosBultos1.cuernosBultos",rand(1));
if (rand(1) > $nivel) setAttr ("cuernosCerrados1.cuernosCerrados",rand(1));
if (rand(1) > $nivel) setAttr ("cuernosDelanteros1.cuernosDelanteros",rand(1));
if (rand(1) > $nivel) setAttr ("cuernosGrandes1.cuernosGrandes" ,rand(1));
if (rand(1) > $nivel) setAttr ("cuernosParabolicos1.cuernosParabolicos" ,rand(1));
if (rand(1) > $nivel) setAttr ("orejasCuernosArriba1.orejasCuernosArriba" ,rand(1));
if (rand(1) > $nivel) setAttr ("orejasRadar1.orejasRadar" ,rand(1));
if (rand(1) > $nivel) setAttr ("orejasTrompeta1.orejasTrompeta" ,rand(1));
if (rand(1) > $nivel) setAttr ("cabezaGrande1.cabezaGrande" ,rand(1));
if (rand(1) > $nivel) setAttr ("cabezaAlargada1.cabezaAlargada" ,rand(1));
if (rand(1) > $nivel) setAttr ("cabezaCuadrada1.cabezaCuadrada" ,rand(1));
if (rand(1) > $nivel) setAttr ("cabezaPequenia1.cabezaPequenia" ,rand(1));
if (rand(1) > $nivel) setAttr ("cascoMilitar1.cascoMilitar" ,rand(1));
if (rand(1) > $nivel) setAttr ("crestaPunk1.crestaPunk",rand(1));
if (rand(1) > $nivel) setAttr ("cuernosCortados1.cuernosCortados",rand(1));
if (rand(1) > $nivel) setAttr ("peloEspinete1.peloEspinete",rand(1));
if (rand(1) > $nivel) setAttr ("cabezaMarsAttacks1.cabezaMarsAttacks" ,rand(1));
if (rand(1) > $nivel) setAttr ("peloCenicero1.peloCenicero" ,rand(1));
if (rand(1) > $nivel) setAttr ("peloCepillo1.peloCepillo",rand(1));
if (rand(1) > $nivel) setAttr ("peloTupe1.peloTupe" ,rand(1));
if (rand(1) > $nivel) setAttr ("cabezaRoswell1.cabezaRoswell" ,rand(1));
if (rand(1) > $nivel) setAttr ("peloTechnopunk1.peloTechnopunk",rand(1));
if (rand(1) > $nivel) setAttr ("peloAfro1.peloAfro",rand(1));
if (rand(1) > $nivel) setAttr ("cabezaAliens1.cabezaAliens",rand(1));
if (rand(1) > $nivel) setAttr ("ojosEstrechos1.ojosEstrechos" ,rand(1));
if (rand(1) > $nivel) setAttr ("ojosNarizEstrecho1.ojosNarizEstrecho",rand(1));
if (rand(1) > $nivel) setAttr ("barbillaConica1.barbillaConica" ,rand(1));
if (rand(1) > $nivel) setAttr ("barbillaCuadrada1.barbillaCuadrada" ,rand(1));
if (rand(1) > $nivel) setAttr ("barbillaSalida1.barbillaSalida",rand(1));
if (rand(1) > $nivel) setAttr ("bocaCorleone1.bocaCorleone" ,rand(1));
if (rand(1) > $nivel) setAttr ("bocaDescolgada1.bocaDescolgada" ,rand(1));
if (rand(1) > $nivel) setAttr ("bocaGrande1.bocaGrande",rand(1));
if (rand(1) > $nivel) setAttr ("cejasRectas1.cejasRectas" ,rand(1));
if (rand(1) > $nivel) setAttr ("cejasTristes1.cejasTristes",rand(1));
if (rand(1) > $nivel) setAttr ("chupandoLimon1.chupandoLimon" ,rand(1));
if (rand(1) > $nivel) setAttr ("mandibulaProminente1.mandibulaProminente" ,rand(1));
if (rand(1) > $nivel) setAttr ("narizAguilenia1.narizAguilenia" ,rand(1));
if (rand(1) > $nivel) setAttr ("narizChata1.narizChata" ,rand(1));
if (rand(1) > $nivel) setAttr ("narizGrande1.narizGrande",rand(1));
if (rand(1) > $nivel) setAttr ("pomulosAltos1.pomulosAltos" ,rand(1));
if (rand(1) > $nivel) setAttr ("pomulosGrandes1.pomulosGrandes" ,rand(1));
if (rand(1) > $nivel) setAttr ("ojeras1.ojeras",rand(1));
if (rand(1) > $nivel) setAttr ("pocaPequenia1.pocaPequenia" ,rand(1));
if (rand(1) > $nivel) setAttr ("caderasEstrechas1.caderasEstrechas",rand(1));
if (rand(1) > $nivel) setAttr ("colaCoxis1.colaCoxis" ,rand(1));
if (rand(1) > $nivel) setAttr ("culoPequenio1.culoPequenio" ,rand(1));
if (rand(1) > $nivel) setAttr ("lateralesGrandes1.lateralesGrandes" ,rand(1));
if (rand(1) > $nivel) setAttr ("espaldasPequenias1.espaldasPequenias" ,rand(1));
if (rand(1) > $nivel) setAttr ("paqueteGrande1.paqueteGrande" ,rand(1));
if (rand(1) > $nivel) setAttr ("culoGrande1.culoGrande",rand(1));
if (rand(1) > $nivel) setAttr ("caderasAnchas1.caderasAnchas" ,rand(1));
```

```
if (rand(1) > $nivel) setAttr ("espaldasAnchas1.espaldasAnchas",rand(1));
if (rand(1) > $nivel) setAttr ("pechoPequenio1.pechoPequenio" ,rand(1));
if (rand(1) > $nivel) setAttr ("torsoGrande1.torsoGrande" ,rand(1));
if (rand(1) > $nivel) setAttr ("barriga1.barriga" ,rand(1));
if (rand(1) > $nivel) setAttr ("chepa1.chepa" ,rand(1));
if (rand(1) > $nivel) setAttr ("camisaArrugasLigeras1.camisaArrugasLigeras" ,rand(1));
if (rand(1) > $nivel) setAttr ("camisaLargaLisa1.camisaLargaLisa" ,rand(1));
if (rand(1) > $nivel) setAttr ("antebrazoGrande1.antebrazoGrande" ,rand(1));
if (rand(1) > $nivel) setAttr ("antebrazoPequenio1.antebrazoPequenio" ,rand(1));
if (rand(1) > $nivel) setAttr ("bicepsPequenios1.bicepsPequenios",rand(1));
if (rand(1) > $nivel) setAttr ("bicepsGrandes1.bicepsGrandes" ,rand(1));
if (rand(1) > $nivel) setAttr ("brazosGrandes1.brazosGrandes" ,rand(1));
if (rand(1) > $nivel) setAttr ("brazosPequenios1.brazosPequenios",rand(1));
if (rand(1) > $nivel) setAttr ("brazosLargos1.brazosLargos" ,rand(1));
if (rand(1) > $nivel) setAttr ("hombrosPequenios1.hombrosPequenios" ,rand(1));
if (rand(1) > $nivel) setAttr ("hombrosGrandes1.hombrosGrandes" ,rand(1));
if (rand(1) > $nivel) setAttr ("homenajeMikeMignola1.homenajeMikeMignola" ,rand(1));
if (rand(1) > $nivel) setAttr ("manosEspadas1.manosEspadas" ,rand(1));
if (rand(1) > $nivel) setAttr ("manosGarras1.manosGarras" ,rand(1));
if (rand(1) > $nivel) setAttr ("manosGrandes1.manosGrandes" ,rand(1));
if (rand(1) > $nivel) setAttr ("manosSartenes1.manosSartenes" ,rand(1));
if (rand(1) > $nivel) setAttr ("puniosCerrados1.puniosCerrados" ,rand(1));
if (rand(1) > $nivel) setAttr ("tricepsGrandes1.tricepsGrandes" ,rand(1));
if (rand(1) > $nivel) setAttr ("tricepsPequenios1.tricepsPequenios" ,rand(1));
if (rand(1) > $nivel) setAttr ("cuchillasWolverine1.cuchillasWolverine" ,rand(1));
if (rand(1) > $nivel) setAttr ("pinchosHombros1.pinchosHombros" ,rand(1));
if (rand(1) > $nivel) setAttr ("botasElficas1.botasElficas" ,rand(1));
if (rand(1) > $nivel) setAttr ("botasMilitares1.botasMilitares",rand(1));
if (rand(1) > $nivel) setAttr ("pantalonesHolgados1.pantalonesHolgados" ,rand(1));
if (rand(1) > $nivel) setAttr ("pantalonesLisos1.pantalonesLisos" ,rand(1));
if (rand(1) > $nivel) setAttr ("piernasPequenias1.piernasPequenias" ,rand(1));
if (rand(1) > $nivel) setAttr ("gemelosAnchos1.gemelosAnchos" ,rand(1));
if (rand(1) > $nivel) setAttr ("gemelosEstrechos1.gemelosEstrechos" ,rand(1));
if (rand(1) > $nivel) setAttr ("muslosAnchos1.muslosAnchos" ,rand(1));
if (rand(1) > $nivel) setAttr ("piesGarras1.piesGarras" ,rand(1));
if (rand(1) > $nivel) setAttr ("piesGrandes1.piesGrandes" ,rand(1));
if (rand(1) > $nivel) setAttr ("pinchosMuslos1.pinchosMuslos" ,rand(1));
if (rand(1) > $nivel) setAttr ("pinchoRodilla1.pinchoRodilla" ,rand(1));
if (rand(1) > $nivel) setAttr ("pinchoTobillo1.pinchoTobillo" ,rand(1));
if (rand(1) > $nivel) setAttr ("muslosEstrechos1.muslosEstrechos",rand(1));
if (rand(1) > $nivel) setAttr ("babuchas1.babuchas" ,rand(1));
if (rand(1) > $nivel) setAttr ("espolonTobillo1.espolonTobillo" ,rand(1));
if (rand(1) > $nivel) setAttr ("pezunia1.pezunia" ,rand(1));
if (rand(1) > $nivel) setAttr ("alto1.alto" ,rand(1));
if (rand(1) > $nivel) setAttr ("enano1.enano" ,rand(1));
if (rand(1) > $nivel) setAttr ("esqueleto1.esqueleto" ,rand(1));
//if (rand(1) > $nivel) setAttr (setAttr "normalizaSinParallel1.normalizaSinParallel"
,rand(1));

}


//-----------------//
//Zona de ejecucion//
//-----------------//

file -f -new;
Ventana1; Ventana2;



// Creation Date: <29-01-06>
//
//*****************************************
//
// Description: Script que configura el
// directorio de trabajo
//
//*****************************************



//DECLARACION DE VARIABLES Y PROCEDIMIENTOS//

global proc modificaDirectorio()
{
```

```
if ( `window -exists ventanaConfig` )
  deleteUI -window ventanaConfig;

string $window = `window -title "Configuración del Path" ventanaConfig`;
string $form = `formLayout`;

//Intentamos abrir el archivo de texto de configuración
int $existe = `filetest -w "C:/hlocal/muniecoconfig.txt"`;
string $path;

if ($existe == 0)
    {
    $fileId=`fopen "C:/hlocal/muniecoconfig.txt" "w"`;
    print "No se ha encontrado el archivo de config. Creamos un archivo nuevo";
    fprint $fileId "Nula\n";
    fclose $fileId;
    }

$fileId=`fopen "C:/hlocal/muniecoconfig.txt" "r"`;
$path = `fgetline $fileId`;
fclose $fileId;


columnLayout -adjustableColumn true;

    text -label ("La ruta actual es esta:\n"+$path+"\n"+"Escribe la nueva ruta:\n" +
"(Escribe / al final y pulsa enter)\n");
    global string $field = "name";
    textField -cc "getValor($field);" $field;

showWindow;

}


//Procedimiento auxiliar que obtiene el directorio escrito por el usuario
global proc getValor(string $field2)
    {
        string $Directorio;
        $Directorio = `textField -q -text $field2`;
        //print $Directorio;
        $fileId=`fopen "C:/hlocal/muniecoconfig.txt" "w"`;
        frewind $fileId;
        fprint $fileId $Directorio;
        fclose $fileId;
        $Directorio = "source \"" + $Directorio + "Scripts/scriptMenuPrincipal.mel\";";
        if ( `button -exists boton` ) deleteUI -control boton;
        button -label "Listo" -command ($Directorio+"deleteUI -window ventanaConfig;")
boton;

    }



//Zona de ejecucion//

modificaDirectorio;


// Creation Date: <29-01-06>
//
//*****************************************
//
// Description: Script lleva el esqueleto
// a donde se hayan colocado las referencias
//
//*****************************************



//Este procecimiento se encarga de llevar los huesos a donde hayamos colocado los joints
de referencia;

//DECLARACION DE VARIABLES Y PROCEDIMIENTOS//
global proc M()
{
```

```
//Inicialmente hacemos un Freeze Transformations a todos los joints
select -r jointMugnecaDer ;
makeIdentity -apply true -t 1 -r 1 -s 1 -n 0;
select -r jointMugnecaIz ;
makeIdentity -apply true -t 1 -r 1 -s 1 -n 0;
select -r jointTobilloDer ;
makeIdentity -apply true -t 1 -r 1 -s 1 -n 0;
select -r jointTobilloIz ;
makeIdentity -apply true -t 1 -r 1 -s 1 -n 0;
select -r jointCabeza ;
makeIdentity -apply true -t 1 -r 1 -s 1 -n 0;
select -r jointRaiz ;
makeIdentity -apply true -t 1 -r 1 -s 1 -n 0;
select -r jointCodoDer ;
makeIdentity -apply true -t 1 -r 1 -s 1 -n 0;
select -r jointCodoIz ;
makeIdentity -apply true -t 1 -r 1 -s 1 -n 0;
select -r jointRodillaDer ;
makeIdentity -apply true -t 1 -r 1 -s 1 -n 0;
select -r jointRodillaIz ;
makeIdentity -apply true -t 1 -r 1 -s 1 -n 0;
select -r jointHombroDer ;
makeIdentity -apply true -t 1 -r 1 -s 1 -n 0;
select -r jointHombroIz ;
makeIdentity -apply true -t 1 -r 1 -s 1 -n 0;


select -r jointCaderaDer ;
makeIdentity -apply true -t 1 -r 1 -s 1 -n 0;
select -r jointCaderaIz ;
makeIdentity -apply true -t 1 -r 1 -s 1 -n 0;
select -r jointEspalda ;
makeIdentity -apply true -t 1 -r 1 -s 1 -n 0;
select -r jointCuello ;
makeIdentity -apply true -t 1 -r 1 -s 1 -n 0;
select -r jointClaviculaDer ;
makeIdentity -apply true -t 1 -r 1 -s 1 -n 0;
select -r jointClaviculaIz ;


float $position[3];

//Hueso Raiz
$position[0] = `getAttr jointRaiz.translateX`;
$position[1] = `getAttr jointRaiz.translateY`;
$position[2] = `getAttr jointRaiz.translateZ`;
select Hips;
move -a $position[0] $position[1] $position[2];

//Espalda
$position[0] = `getAttr jointEspalda.translateX`;
$position[1] = `getAttr jointEspalda.translateY`;
$position[2] = `getAttr jointEspalda.translateZ`;
select Chest;
move -a $position[0] $position[1] $position[2];

//Cuello
$position[0] = `getAttr jointCuello.translateX`;
$position[1] = `getAttr jointCuello.translateY`;
$position[2] = `getAttr jointCuello.translateZ`;
select Neck;
move -a $position[0] $position[1] $position[2];


//Cabeza
$position[0] = `getAttr jointCabeza.translateX`;
$position[1] = `getAttr jointCabeza.translateY`;
$position[2] = `getAttr jointCabeza.translateZ`;
select Head;
move -a $position[0] $position[1] $position[2];


//Clavicula Izquierda
$position[0] = `getAttr jointClaviculaIz.translateX`;
$position[1] = `getAttr jointClaviculaIz.translateY`;
$position[2] = `getAttr jointClaviculaIz.translateZ`;
select LeftCollar;
```

```
move -a $position[0] $position[1] $position[2];

//Clavicula Derecha
$position[0] = `getAttr jointClaviculaDer.translateX`;
$position[1] = `getAttr jointClaviculaDer.translateY`;
$position[2] = `getAttr jointClaviculaDer.translateZ`;
select RightCollar;
move -a $position[0] $position[1] $position[2];


//Hombro Izquierdo
$position[0] = `getAttr jointHombroIz.translateX`;
$position[1] = `getAttr jointHombroIz.translateY`;
$position[2] = `getAttr jointHombroIz.translateZ`;
select LeftUpArm;
move -a $position[0] $position[1] $position[2];

//Hombro Derecho
$position[0] = `getAttr jointHombroDer.translateX`;
$position[1] = `getAttr jointHombroDer.translateY`;
$position[2] = `getAttr jointHombroDer.translateZ`;
select RightUpArm;
move -a $position[0] $position[1] $position[2];

//Codo Izquierdo
$position[0] = `getAttr jointCodoIz.translateX`;
$position[1] = `getAttr jointCodoIz.translateY`;
$position[2] = `getAttr jointCodoIz.translateZ`;
select LeftLowArm;
move -a $position[0] $position[1] $position[2];

//Codo Derecho
$position[0] = `getAttr jointCodoDer.translateX`;
$position[1] = `getAttr jointCodoDer.translateY`;
$position[2] = `getAttr jointCodoDer.translateZ`;
select RightLowArm;
move -a $position[0] $position[1] $position[2];

//Muñeca Izquierda
$position[0] = `getAttr jointMugnecaIz.translateX`;
$position[1] = `getAttr jointMugnecaIz.translateY`;
$position[2] = `getAttr jointMugnecaIz.translateZ`;
select LeftHand;
move -a $position[0] $position[1] $position[2];

//Muñeca Derecha
$position[0] = `getAttr jointMugnecaDer.translateX`;
$position[1] = `getAttr jointMugnecaDer.translateY`;
$position[2] = `getAttr jointMugnecaDer.translateZ`;
select RightHand;
move -a $position[0] $position[1] $position[2];


//Cadera Izquierda
$position[0] = `getAttr jointCaderaIz.translateX`;
$position[1] = `getAttr jointCaderaIz.translateY`;
$position[2] = `getAttr jointCaderaIz.translateZ`;
select LeftUpLeg;
move -a $position[0] $position[1] $position[2];

//Cadera Derecha
$position[0] = `getAttr jointCaderaDer.translateX`;
$position[1] = `getAttr jointCaderaDer.translateY`;
$position[2] = `getAttr jointCaderaDer.translateZ`;
select RightUpLeg;
move -a $position[0] $position[1] $position[2];

//Rodilla Izquierda
$position[0] = `getAttr jointRodillaIz.translateX`;
$position[1] = `getAttr jointRodillaIz.translateY`;
$position[2] = `getAttr jointRodillaIz.translateZ`;
select LeftLowLeg;
move -a $position[0] $position[1] $position[2];

//Rodilla Derecha
$position[0] = `getAttr jointRodillaDer.translateX`;
$position[1] = `getAttr jointRodillaDer.translateY`;
```

```
$position[2] = `getAttr jointRodillaDer.translateZ`;
select RightLowLeg;
move -a $position[0] $position[1] $position[2];

//Tobillo Izquierdo
$position[0] = `getAttr jointTobilloIz.translateX`;
$position[1] = `getAttr jointTobilloIz.translateY`;
$position[2] = `getAttr jointTobilloIz.translateZ`;
select LeftFoot;
move -a $position[0] $position[1] $position[2];

//Tobillo Derecho
$position[0] = `getAttr jointTobilloDer.translateX`;
$position[1] = `getAttr jointTobilloDer.translateY`;
$position[2] = `getAttr jointTobilloDer.translateZ`;
select RightFoot;
move -a $position[0] $position[1] $position[2];

}

//Zona de ejecucion//

M;


// Creation Date: <29-01-06>
//
//*****************************************
//
// Description: Script generador de la
// ventana de ayuda
//
//*****************************************



//DECLARACION DE VARIABLES Y PROCEDIMIENTOS//
global proc M()
{

if ( `window -exists ventanaAyuda` )
  deleteUI -window ventanaAyuda;

string $window = `window -title "Ventana de Ayuda" ventanaAyuda`;

rowColumnLayout -numberOfColumns 2 -w 500;
text -label " " -w 500;
text -label " " -w 500;
text -label " INSTRUCCIONES" -w 500;
text -label "DE  USO  EN:" -w 500;
text -label " Ayuda.html" -w 500;

showWindow;
}

//Zona de ejecucion//

M;


// Creation Date: <29-01-06>
//
//*****************************************
//
// Description: Script que elimina las
// capas de textura
//
//*****************************************



delete "Barba2";
delete "pelo3_alpha";
delete "blendColors1";
delete "place2dTexture41";
delete "place2dTexture42";
delete "place2dTexture43";
delete "place2dTexture48";
```

```
delete "place2dTexture49";
delete "place2dTexture129";
delete "place2dTexture131";
delete "layeredTexture1";
delete "ramp1";
delete "place2dTexture6";
delete "Barba2_alpha";
delete "armor1_alpha";
delete "armor2_alpha";
delete "armor_1";
delete "armor_2";
delete "barba1";
delete "barba1_alpha";
delete "barba3";
delete "barba3_alpha";
delete "barba4";
delete "barba4_alpha";
delete "bigote1";
delete "bigote2";
delete "bigote2_alpha";
delete "bigote3";
delete "bigote3_alpha";
delete "botas";
delete "cal_cor_alpha";
delete "cal_lar_alpha";
delete "calcetines_cortos";
delete "calcetines_cortos2";
delete "calcetines_largos";
delete "calcetines_largos2";
delete "cam_corb";
delete "cam_corb2";
delete "cam_corb3";
delete "cam_corb4";
delete "cam_corb_alpha";
delete "cam_lar";
delete "cam_lar1";
delete "cam_lar2";
delete "cam_lar3";
delete "cam_lar4";
delete "cam_lar_2";
delete "cam_lar_3";
delete "cam_lar_4";
delete "cam_lar_alpha";
delete "cam_lar_alpha1";
delete "cam_tir";
delete "cam_tir_2";
delete "cam_tir_3";
delete "cam_tir_4";
delete "cam_tir_5";
delete "cam_tir_alpha";
delete "camiseta";
delete "camiseta2";
delete "camiseta_3";
delete "camiseta_4";
delete "camiseta_5";
delete "camiseta_6";
delete "camiseta_alpha";
delete "camuf";
delete "camuf2";
delete "camuf3";
delete "camuf4";
delete "camuf_alpha";
delete "casco";
delete "casco_alpha";
delete "cejas1";
delete "cejas1_alpha";
delete "cicatriz";
delete "cicatriz1";
delete "corbata";
delete "corbata2";
delete "corbata3";
delete "corbata4";
delete "corbata5";
delete "corbata_alpha";
delete "cuernos";
delete "cuernos_alpha";
delete "gafas1alpha";
```

```
delete "gafassol1";
delete "gafassol2";
delete "gafassol2_alpha";
delete "malla";
delete "malla2";
delete "malla3";
delete "malla_alpha";
delete "malla_ref_alpha";
delete "malla_refor";
delete "pan_cor";
delete "pan_cor_2";
delete "pan_cor_3";
delete "pan_cor_4";
delete "pan_cor_5";
delete "pan_cor_alpha";
delete "panta_alpha";
delete "pantalones";
delete "pantalones2";
delete "pantalones3";
delete "pantalones4";
delete "pantalones5";
delete "pasamon";
delete "pasamon_alpha";
delete "pelo1";
delete "pelo1_alpha";
delete "pelo2";
delete "pelo2_alpha";
delete "pelo3";
delete "perilla1";
delete "perilla1_alpha";
delete "perilla2";
delete "perilla2_alpha";
delete "piel1";
delete "piel2";
delete "piel3";
delete "piel4";
delete "piel5";
delete "piel6";
delete "piel7";
delete "piel8";
delete "piel9";
delete "piel10";
delete "piel11";
delete "piel12";
delete "piel13";
delete "piel14";
delete "piel15";
delete "piel16";
delete "piel17";
delete "piel18";
delete "piel19";
delete "prot";
delete "prot_alpha";
delete "spacial1";
delete "spacial2";
delete "spacial3";
delete "spacial4";
delete "spacial_alpha";
delete "spidey";
delete "spidey_negro";
delete "swat";
delete "swat_alpha";
delete "zapatos";
delete "zapatos2";
delete "zapatos3";
delete "zapatos_alpha";
delete "place2dTexture7";
delete "place2dTexture8";
delete "place2dTexture9";
delete "place2dTexture10";
delete "place2dTexture11";
delete "place2dTexture12";
delete "place2dTexture13";
delete "place2dTexture14";
delete "place2dTexture15";
delete "place2dTexture16";
delete "place2dTexture17";
```

```
delete "place2dTexture18";
delete "place2dTexture19";
delete "place2dTexture20";
delete "place2dTexture21";
delete "place2dTexture22";
delete "place2dTexture23";
delete "place2dTexture24";
delete "place2dTexture25";
delete "place2dTexture26";
delete "place2dTexture27";
delete "place2dTexture28";
delete "place2dTexture29";
delete "place2dTexture30";
delete "place2dTexture32";
delete "place2dTexture33";
delete "place2dTexture34";
delete "place2dTexture35";
delete "place2dTexture36";
delete "place2dTexture37";
delete "place2dTexture38";
delete "place2dTexture39";
delete "place2dTexture40";
delete "place2dTexture44";
delete "place2dTexture45";
delete "place2dTexture46";
delete "place2dTexture47";
delete "place2dTexture50";
delete "place2dTexture51";
delete "place2dTexture52";
delete "place2dTexture53";
delete "place2dTexture54";
delete "place2dTexture55";
delete "place2dTexture56";
delete "place2dTexture57";
delete "place2dTexture58";
delete "place2dTexture59";
delete "place2dTexture60";
delete "place2dTexture61";
delete "place2dTexture62";
delete "place2dTexture63";
delete "place2dTexture64";
delete "place2dTexture65";
delete "place2dTexture66";
delete "place2dTexture67";
delete "place2dTexture68";
delete "place2dTexture69";
delete "place2dTexture70";
delete "place2dTexture71";
delete "place2dTexture72";
delete "place2dTexture73";
delete "place2dTexture74";
delete "place2dTexture75";
delete "place2dTexture76";
delete "place2dTexture77";
delete "place2dTexture78";
delete "place2dTexture79";
delete "place2dTexture80";
delete "place2dTexture81";
delete "place2dTexture82";
delete "place2dTexture83";
delete "place2dTexture84";
delete "place2dTexture85";
delete "place2dTexture86";
delete "place2dTexture87";
delete "place2dTexture88";
delete "place2dTexture89";
delete "place2dTexture90";
delete "place2dTexture91";
delete "place2dTexture92";
delete "place2dTexture93";
delete "place2dTexture94";
delete "place2dTexture95";
delete "place2dTexture96";
delete "place2dTexture97";
delete "place2dTexture98";
delete "place2dTexture99";
delete "place2dTexture100";
```

```
delete "place2dTexture101";
delete "place2dTexture102";
delete "place2dTexture103";
delete "place2dTexture104";
delete "place2dTexture105";
delete "place2dTexture106";
delete "place2dTexture107";
delete "place2dTexture108";
delete "place2dTexture109";
delete "place2dTexture110";
delete "place2dTexture111";
delete "place2dTexture112";
delete "place2dTexture113";
delete "place2dTexture114";
delete "place2dTexture115";
delete "place2dTexture116";
delete "place2dTexture117";
delete "place2dTexture118";
delete "place2dTexture119";
delete "place2dTexture120";
delete "place2dTexture121";
delete "place2dTexture122";
delete "place2dTexture123";
delete "place2dTexture124";
delete "place2dTexture125";
delete "place2dTexture126";
delete "place2dTexture127";
delete "place2dTexture128";
delete "place2dTexture130";
delete "place2dTexture132";
delete "place2dTexture133";
delete "place2dTexture134";
delete "place2dTexture135";
delete "place2dTexture136";
delete "place2dTexture137";
delete "place2dTexture138";
delete "place2dTexture139";
delete "place2dTexture140";
delete "place2dTexture141";
delete "place2dTexture142";
delete "place2dTexture143";
delete "place2dTexture144";
delete "place2dTexture145";
delete "place2dTexture146";
delete "place2dTexture147";
delete "place2dTexture148";
delete "place2dTexture149";
delete "place2dTexture150";
delete "place2dTexture151";
select camara1; delete;
select camara2; delete;
select luz; delete;



// Creation Date: <29-01-06>
//
//**************************************
//
// Description: Script que elimina las
// shapes de la escena
//
//**************************************



select -r barbillaCuadrada ;
select -add barbillaConica ;
select -add abdominales ;
select -add barbillaSalida ;
select -add bicepsGrandes ;
select -add bocaCorleone ;
select -add bocaDescolgada ;
select -add bocaGrande ;
select -add botasElficas ;
select -add caderasEstrechas ;
select -add cejasRectas ;
```

```
select -add cejasTristes ;
select -add chupandoLimon ;
select -add colaCoxis ;
select -add culoPequenio ;
select -add lateralesGrandes ;
select -add espaldasPequenias ;
select -add espolonTobillo ;
select -add gemelosAnchos ;
select -add gemelosEstrechos ;
select -add mandibulaProminente ;
select -d mandibulaProminente ;
select -add mandibulaProminente ;
select -add muslosAnchos ;
select -add narizAguilenia ;
select -add narizChata ;
select -add narizGrande ;
select -add orejasElficas ;
select -add orejasGordas ;
select -add orejasGrandes ;
select -add orejasNormales ;
select -add orejasPequenias ;
select -add orejasYoda ;
select -add paqueteGrande ;
select -add pezunia ;
select -add piesGarras ;
select -add piesGrandes ;
select -add pinchosMuslos ;
select -add pinchoRodilla ;
select -add pinchosHombros ;
select -add pinchoTobillo ;
select -add pomulosAltos ;
select -add pomulosGrandes ;
select -add ojeras ;
select -add pocaPequenia ;
select -add muslosEstrechos ;
select -add culoGrande ;
select -add caderasAnchas ;
select -add babuchas ;
select -add cuelloCorto ;
select -add cuelloEstrecho ;
select -add cuelloLargo ;
select -add espaldasAnchas ;
select -add pechoPequenio ;
select -add torsoGrande ;
select -add barriga ;
select -add chepa ;
select -add cuernosAtras ;
select -add cuernosBajos ;
select -add cuernosBolas ;
select -add cuernosBultos ;
select -add cuernosCerrados ;
select -add cuernosDelanteros ;
select -add cuernosGrandes ;
select -add cuernosParabolicos ;
select -add orejasCuernosArriba ;
select -add orejasRadar ;
select -add orejasTrompeta ;
select -add cabezaGrande ;
select -add antebrazoGrande ;
select -add antebrazoPequenio ;
select -add bicepsPequenios ;
select -add botasMilitares ;
select -add brazosGrandes ;
select -add brazosLargos ;
select -add brazosPequenios ;
select -add cabezaAlargada ;
select -add cabezaCuadrada ;
select -add cabezaPequenia ;
select -add camisaArrugasLigeras ;
select -add cascoMilitar ;
select -add crestaPunk ;
select -add cuchillasWolverine ;
select -add cuernosCortados ;
select -add peloEspinete ;
select -add hombrosPequenios ;
select -add hombrosGrandes ;
select -add homenajeMikeMignola ;
```

```
select -add manosEspadas ;
select -add manosGarras ;
select -add manosGrandes ;
select -add manosSartenes ;
select -add cabezaMarsAttacks ;
select -add pantalonesHolgados ;
select -add pantalonesLisos ;
select -add peloCenicero ;
select -add peloCepillo ;
select -add piernasPequenias ;
select -add puniosCerrados ;
select -add tricepsGrandes ;
select -add tricepsPequenios ;
select -add peloTupe ;
select -add ojosEstrechos ;
select -add ojosNarizEstrecho ;
select -add cuelloAncho ;
select -add cabezaRoswell ;
select -add alto ;
select -add peloTechnopunk ;
select -add enano ;
select -add peloAfro ;
select -add esqueleto ;
select -add camisaLargaLisa ;
select -add cabezaAliens ;
select -add normalizaSinParallel ;
delete;



// Creation Date: <29-01-06>
//
//*****************************************
//
// Description: Script para crear los joints
//
//*****************************************



//DECLARACION DE VARIABLES Y PROCEDIMIENTOS//
global proc joints()
{

//Ponemos la cámara en modo X-Ray + Wireframe para que se vea la posición de los puntos
de referencia
modelEditor -e -xray 1 modelPanel4;
modelEditor -e -wos 1 modelPanel4;


//Hacemos visibles los joints
setAttr "jointMugnecaDer.visibility" 1;
setAttr "jointMugnecaIz.visibility" 1;
setAttr "jointTobilloDer.visibility" 1;
setAttr "jointTobilloIz.visibility" 1;
setAttr "jointCodoDer.visibility" 1;
setAttr "jointCodoIz.visibility" 1;
setAttr "jointHombroDer.visibility" 1;
setAttr "jointHombroIz.visibility" 1;
setAttr "jointRodillaDer.visibility" 1;
setAttr "jointRodillaIz.visibility" 1;
setAttr "jointRaiz.visibility" 1;
setAttr "jointCabeza.visibility" 1;
setAttr "jointCaderaDer.visibility" 1;
setAttr "jointCaderaIz.visibility" 1;
setAttr "jointClaviculaDer.visibility" 1;
setAttr "jointClaviculaIz.visibility" 1;
setAttr "jointCuello.visibility" 1;
setAttr "jointEspalda.visibility" 1;

// Colocamos los huesos con respecto a los vertices
float $pos_a[] = `pointPosition Dummy.vtx[1016]`;
float $pos_b[] = `pointPosition Dummy.vtx[972]`;
float $pos[3];
$pos[0]=($pos_a[0]+$pos_b[0])/2;
$pos[1]=($pos_a[1]+$pos_b[1])/2;
$pos[2]=($pos_a[2]+$pos_b[2])/2;
```

```
select jointCabeza;
move -a $pos[0] $pos[1] $pos[2];


$pos_a = `pointPosition Dummy.vtx[32]`;
$pos_b = `pointPosition Dummy.vtx[39]`;
$pos[0]=($pos_a[0]+$pos_b[0])/2;
$pos[1]=($pos_a[1]+$pos_b[1])/2;
$pos[2]=($pos_a[2]+$pos_b[2])/2;

select jointCuello;
move -a $pos[0] $pos[1] $pos[2];


$pos_a = `pointPosition Dummy.vtx[208]`;
$pos_b = `pointPosition Dummy.vtx[106]`;
$pos[0]=($pos_a[0]+$pos_b[0])/2;
$pos[1]=($pos_a[1]+$pos_b[1])/2;
$pos[2]=($pos_a[2]+$pos_b[2])/2;

select jointEspalda;
move -a $pos[0] $pos[1] $pos[2];


$pos_a = `pointPosition Dummy.vtx[602]`;
$pos_b = `pointPosition Dummy.vtx[692]`;
$pos[0]=($pos_a[0]+$pos_b[0])/2;
$pos[1]=($pos_a[1]+$pos_b[1])/2;
$pos[2]=($pos_a[2]+$pos_b[2])/2;

select jointRaiz;
move -a $pos[0] $pos[1] $pos[2];


$pos_a = `pointPosition Dummy.vtx[654]`;
$pos_b = `pointPosition Dummy.vtx[619]`;
$pos[0]=($pos_a[0]+$pos_b[0])/2;
$pos[1]=($pos_a[1]+$pos_b[1])/2;
$pos[2]=($pos_a[2]+$pos_b[2])/2;

select jointCaderaDer;
move -a $pos[0] $pos[1] $pos[2];


$pos_a = `pointPosition Dummy.vtx[1925]`;
$pos_b = `pointPosition Dummy.vtx[1958]`;
$pos[0]=($pos_a[0]+$pos_b[0])/2;
$pos[1]=($pos_a[1]+$pos_b[1])/2;
$pos[2]=($pos_a[2]+$pos_b[2])/2;

select jointCaderaIz;
move -a $pos[0] $pos[1] $pos[2];


$pos_a = `pointPosition Dummy.vtx[746]`;
$pos_b = `pointPosition Dummy.vtx[752]`;
$pos[0]=($pos_a[0]+$pos_b[0])/2;
$pos[1]=($pos_a[1]+$pos_b[1])/2;
$pos[2]=($pos_a[2]+$pos_b[2])/2;

select jointRodillaDer;
move -a $pos[0] $pos[1] $pos[2];


$pos_a = `pointPosition Dummy.vtx[2052]`;
$pos_b = `pointPosition Dummy.vtx[2046]`;
$pos[0]=($pos_a[0]+$pos_b[0])/2;
$pos[1]=($pos_a[1]+$pos_b[1])/2;
$pos[2]=($pos_a[2]+$pos_b[2])/2;

select jointRodillaIz;
move -a $pos[0] $pos[1] $pos[2];


$pos_a = `pointPosition Dummy.vtx[2181]`;
$pos_b = `pointPosition Dummy.vtx[2163]`;
```

```
$pos[0]=($pos_a[0]+$pos_b[0])/2;
$pos[1]=($pos_a[1]+$pos_b[1])/2;
$pos[2]=($pos_a[2]+$pos_b[2])/2;

select jointTobilloIz;
move -a $pos[0] $pos[1] $pos[2];


$pos_a = `pointPosition Dummy.vtx[863]`;
$pos_b = `pointPosition Dummy.vtx[881]`;
$pos[0]=($pos_a[0]+$pos_b[0])/2;
$pos[1]=($pos_a[1]+$pos_b[1])/2;
$pos[2]=($pos_a[2]+$pos_b[2])/2;

select jointTobilloDer;
move -a $pos[0] $pos[1] $pos[2];


$pos_a = `pointPosition Dummy.vtx[253]`;
$pos_b = `pointPosition Dummy.vtx[117]`;
$pos[0]=($pos_a[0]+$pos_b[0])/2;
$pos[1]=($pos_a[1]+$pos_b[1])/2;
$pos[2]=($pos_a[2]+$pos_b[2])/2;

select jointClaviculaDer;
move -a $pos[0] $pos[1] $pos[2];


$pos_a = `pointPosition Dummy.vtx[1450]`;
$pos_b = `pointPosition Dummy.vtx[1579]`;
$pos[0]=($pos_a[0]+$pos_b[0])/2;
$pos[1]=($pos_a[1]+$pos_b[1])/2;
$pos[2]=($pos_a[2]+$pos_b[2])/2;

select jointClaviculaIz;
move -a $pos[0] $pos[1] $pos[2];


$pos_a = `pointPosition Dummy.vtx[280]`;
$pos_b = `pointPosition Dummy.vtx[251]`;
$pos[0]=($pos_a[0]+$pos_b[0])/2;
$pos[1]=($pos_a[1]+$pos_b[1])/2;
$pos[2]=($pos_a[2]+$pos_b[2])/2;

select jointHombroDer;
move -a $pos[0] $pos[1] $pos[2];


$pos_a = `pointPosition Dummy.vtx[1602]`;
$pos_b = `pointPosition Dummy.vtx[1577]`;
$pos[0]=($pos_a[0]+$pos_b[0])/2;
$pos[1]=($pos_a[1]+$pos_b[1])/2;
$pos[2]=($pos_a[2]+$pos_b[2])/2;

select jointHombroIz;
move -a $pos[0] $pos[1] $pos[2];


$pos_a = `pointPosition Dummy.vtx[80]`;
$pos_b = `pointPosition Dummy.vtx[122]`;
$pos[0]=($pos_a[0]+$pos_b[0])/2;
$pos[1]=($pos_a[1]+$pos_b[1])/2;
$pos[2]=($pos_a[2]+$pos_b[2])/2;

select jointCodoDer;
move -a $pos[0] $pos[1] $pos[2];


$pos_a = `pointPosition Dummy.vtx[1416]`;
$pos_b = `pointPosition Dummy.vtx[1455]`;
$pos[0]=($pos_a[0]+$pos_b[0])/2;
$pos[1]=($pos_a[1]+$pos_b[1])/2;
$pos[2]=($pos_a[2]+$pos_b[2])/2;

select jointCodoIz;
move -a $pos[0] $pos[1] $pos[2];
```

```
$pos_a = `pointPosition Dummy.vtx[479]`;
$pos_b = `pointPosition Dummy.vtx[484]`;
$pos[0]=($pos_a[0]+$pos_b[0])/2;
$pos[1]=($pos_a[1]+$pos_b[1])/2;
$pos[2]=($pos_a[2]+$pos_b[2])/2;

select jointMugnecaDer;
move -a $pos[0] $pos[1] $pos[2];


$pos_a = `pointPosition Dummy.vtx[1792]`;
$pos_b = `pointPosition Dummy.vtx[1797]`;
$pos[0]=($pos_a[0]+$pos_b[0])/2;
$pos[1]=($pos_a[1]+$pos_b[1])/2;
$pos[2]=($pos_a[2]+$pos_b[2])/2;

select jointMugnecaIz;
move -a $pos[0] $pos[1] $pos[2];

}

//Zona de ejecucion//

joints;



// Creation Date: <29-01-06>
//
//*****************************************
//
// Description: Script para hacer el pesado
// automático del esqueleto a la maya
//
//*****************************************



//DECLARACION DE VARIABLES Y PROCEDIMIENTOS//
global proc PesadoAutomatico()
{
//Volvemos al modo normal de trabajar (quitamos modos X-ray y Wireframe)
modelEditor -e -xray 0 modelPanel4;
modelEditor -e -wos 0 modelPanel4;

select -r Hips ;
select -add Dummy ;
newSkinCluster "-mi 2 -dr 4";


select -r jointMugnecaDer ;
delete;
select -r jointMugnecaIz ;
delete;
select -r jointTobilloDer ;
delete;
select -r jointTobilloIz ;
delete;
select -r jointCabeza ;
delete;
select -r jointRaiz ;
delete;
select -r jointCodoDer ;
delete;
select -r jointCodoIz ;
delete;
select -r jointRodillaDer ;
delete;
select -r jointRodillaIz ;
delete;
select -r jointHombroDer ;
delete;
select -r jointHombroIz ;
delete;
select -r jointCaderaDer ;
delete;
```

```
select -r jointCaderaIz ;
delete;
select -r jointEspalda ;
delete;
select -r jointCuello ;
delete;
select -r jointClaviculaDer ;
delete;
select -r jointClaviculaIz ;
delete;


}

//Zona de ejecucion//

PesadoAutomatico;
```